UNIVERSITY OF OSLO
Department of Informatics

A QoE Framework for Web Services

Master thesis

Mei Yang

Network and System Administration

Oslo University College

Spring 2011

A QoE Framework for Web Services

Mei Yang

Network and System Administration
Oslo University College

Spring 2011

Acknowledgements

I would like to express my gratitude to the following people:

- My supervisor Kyrre Matthias Begnum for his initiating idea of this project, for his support, patience and ingenuity in guiding me on the process and shaping the final implementation of this project.
- The master program committee for their positive attitude and support during the mid-term presentation and application process.
- Mark Burgess for his macroscopic instructions and detailed suggests.
- Hårek Haugerud for good advices during the presentation discussion.
- My friend, Changbing Ren for his technical support locating in Firefox's exension.
- My classmates for fruitful discussions during the whole process of the master program.
- My friends for their suggests, assistance of testing the project and feedback about the questionnaire.
- Last but not least, my family for their long-time supports and continuous love.

Abstract

This thesis aims to discover a relatively new phenomenon QoE. On the premise that the users' perspectives on a certain service are worth to investigate, it is believed that QoE can have explication for the relevant service's quality to some extent. Consequently, a framework on QoE is a demand for this investigation. Meanwhile, an unambiguous service type, which is a web service, is specified as the researching object of QoE framework. Until now, the general and holistic boundary and direction are clearly declared.
Furthermore, a feedback system is achieved for acquiring users' perspectives on web services. An Apache server is for processing these perspectives and offering them to web services vendors by an API. An typical example of how a service vendor interacts with QoE result is demonstrated.
Finally, it proves that it is possbile for a QoE implementation.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Resources management is a big issue in IT industry, which aims to find an efficient and effective solution of assigning resources. However, since the Internet has a exponential grows in scale with the number of websites' increment during the recent few years, the insufficiency of Internet resources happens at times because of the intense competition of resources between Internet users. It is believed that the resources optimization for a website can satisfy more users and ensure the accessibility.

Quality is a significant aspect for any kinds of services. And service providers should define and evaluate their service quality clearly. This also apples to web services which are in the context of information technologies. Traditionally, a web service's quality is measured by QoS(Quality of Service) metrics. For instance, BER(Bit Error Ratio), maximum and mean throughput rate, reliably, priority and other factors specific to each service can be used to quantify the current web service quality.[1] QoS determines the service usability and utility, both of which influence the popularity of the service.[31]

The measurements of Quality of Service metrics can efficiently in control of a web service's quality, which can be guaranteed to a certain extent. And the further adjustments are possible for higher quality of a web service by service provider. Despite of the effects by QoS, it omits end users' perceptions about web services. In other words, it is believed that the end users expectations should be considered when providing a web service. Consequently, here we consider another alternative way for measuring web services based on the users' experiences. Still, both QoS and QoE are devoting to evaluating the current web services so as to provide opinions for services adjustments, the objectives of them are in common on this point.

Quality of Experience(QoE) is a relatively new term which measures how well a service is satisfied the end user's requirements. QoE and QoS are related with each other, though still have difference. As for a web service, its charac-

teristics can be evaluated by users' overall satisfaction. In other words, users' feedbacks about web services can be metrics of evaluating its quality and conduct the service providers to adjust internally in order to satisfy users better. One possible approach of adjustments refers to the resource management, optimizing the resources based on QoE in real practice.

1.2 Problem statement

As QoE is still not matured, it makes sence to put it in real practical implementation so as to understand this concept of QoE. Since the concept of QoE is a bit general, it is specified in the field of web services in this project.
The following problem statement was formulated for this project: *Design and implement a user-centric QoE framework for evaluation of web services.*

A Web service is an inter medium that connects and mediates service providers with clients.

QoE refers to the users' perspectives about the web services compared with their expectations. For example, the network speed and delay when browsing a website will affect the users' experience. When most of the users of the same website feel pleasant on it, it reveals that QoE of this website is high relatively by supplying good experiences to these users.

Framework is a technical approach for QoE in this paper. The first step is to design the framework, followed by implementation. It has three properties which will be the guidelines for the two steps.

* Intuitive: Users express their feelings about the current web service in an intuitive way; the web service providers can attain these feelings also in an intuitive way.
* Ease of use: On one hand, for users' convenience, the process of expressing their perspectives should be as simple as possible. On the other hand, service vendors can obtain these corresponding perspectives effortlessly.
* Timeliness: Users submit the immediate feedbacks. Meanwhile, it is possible for the web service providers to obtain these experience feedbacks in a specific time period.

1.3 Thesis Outline

The paper will be structured as the following manner: Introduction gives the overview of this topic and states the problem solved. Chapter 2 gives the background of QoS, QoE and related technical concepts and relevant software explanations. Chapter 3 tells the methodology of the project, containing the justification of choosing web browser's add-on for implementation of QoE, but

not other alternative ways. Both Chapter 4 and Chapter 5 are result parts: the former discusses architecture of QoE implementation and functionalities of each sub-component in between; the latter one is the implementation of the former part to prove that these sub-systems can collaborate with each other and work. Also a small test is demonstrated. Chapter 6 is for discuss the work having been done and states improvements and possible future work of this topic. Chapter 7 summarizes the whole project.

Chapter 2

Background

This chapter first gives introductions to QoS and QoE. Existed measurements methods of QoE, the related work on QoE are stated. A pre-study of all possible reasons for a bad QoE is discussed.
After that comes all the prerequisites of implementing Firefox's extension and how the extension functions.
The last part is the explanations of other technologies involved in this project.

2.1 QoS

In the field of telephony, QoS was defined by ITU as "the collective effect of service performance which determine the degree of satisfaction of a user of the service". [22]

In the field of computer networking, QoS refers to a resource reservation mechanism.[17] The goal of QoS is to provide guarantees on the ability of a network to deliver predictable results. Elements of network performance within the scope of QoS often include availability (uptime), bandwidth (throughput), latency (delay), and error rate.[2]

Quality of service is a combination of several qualities or properties of a service, such as:

- Field 1: Availability is the percentage of time that a service is operating.
- Field 2: Security properties include the existence and type of authentication mechanisms the service offers, confidentiality and data integrity of messages exchanged, non repudiation of requests or messages, and resilience to denial-of service attacks.
- Field 3: Response time is the time a service takes to respond to various types of requests. Response time is a function of load intensity, which can be measured in terms of arrival rates (such as requests per second) or number of concurrent requests. QoS takes into account not only the

average response time, but also the percentile (95th percentile, for example) of the response time.

- Field 4: Throughput is the rate at which a service can process requests. QoS measures can include the maximum throughput or a function that describes how throughput varies with load intensity.[18]

Historically, the concept of QoS applied to IP-based web services is highly associated with the network and systems themselves. QoS is typically understood as a measure of performance from the network perspective at the packet level. It represents the set of parameters, technologies and mechanisms that should be implemented in the network infrastructure in order to meet the applications performance requirements.
One example is the definition of an end-to-end QoS multimedia in ITU G.1010 [27] that mostly focuses on network parameters instead of user requirements.[30]

In addition, QoS can be implemented over a best-effort network in assumption that the Internet Resources are sufficient for the expected peak traffic load.

2.2 QoE

QoE is a general concept, which is defined in various statements throughout the current papers and publications. For example, in [34], QoE is defined as "the user's perceived experience of what is being presented by the Application Layer, where the application layer acts as a user interface front-end that presents the overall result of the individual Quality of Services". In [32] QoE is defined as "the characteristics of the sensations, perceptions, and opinions of people as they interact with their environments. These characteristics can be pleasing and enjoyable, or displeasing and frustrating". In [33], QoE is defined as "the totality of the Quality of Service mechanisms, provided to ensure smooth transmission of audio and video over IP networks". In [20], "QoE is the overall performance of a system from the point of view of the users. QoE is a measure of an end-to-end performance levels at the user perspective and an indicator of how well this system meets the user needs". In [20], A distinction between QoS and QoE in a sense that QoE stands on the user perspective while QoS pays respect to the network perspective.

Anyhow, the definition or concept of QoE is still abstract. In system administration engineer, we consider concrete performance measures of it since QoE always gives a more subjective impression of us. Soft approaches can be a good choice.

2.2.1 Subjective factors of QoE

There exist so many subjective factors when people judge QoE of surfing the Internet, which makes the measurement of QoE especially when being compared with QoS. In another way, QoS works more on the objectivity, whereas

QoE cares subjective entity's perceptions.

Expectations from various users could be distinct for many causes.

- Users' preferences are subjective for evaluating whether good or not of a website's service. The same network speed can be acceptable by some of the users, whereas rejected by some others.

- Experience is also content-related. On one hand, the personal situation is a inner factor of QoE. On the other hand, the noisy or uncomfortable environment may influence the users' judgement of QOE.

- Tolerance and patience are two critical characteristics which have effect on personal decision grading of QoE.

- On psychological aspect, individual emotions can be interference factors for Quality of Experience.

2.2.2 Probable causes for a bad QoE of web services

Many parties can affect QoE. However, the failures of any of these parties will decrease the Quality of Experience.

- Web Servers' Providers' Problems

  On the hardware's consideration, there can be plenty causes. For example, the aging of network equipments for running hour after hour may lead to the final failure of the facilities. Unfortunately, the data center supporting a website may suffer a physical damage which affects the operation of the site. As for software's aspect, the website's bugs under the codes, the malicious attacks from outside, the failures from DNS servers or some other application servers can result in the crash of a web service provider. Certain software components which are prerequisites for running the website, like jdk, tomcat, may have faults that leading to the website's terrible performance. Also, these facilities can get crashed for restarting them after an inappropriate configuration or some other reasons.

  The possibility of human errors, like the unplugging of power or cable connections, should be considered as well.

- Internet Servers Provider' Problems
  The Internet Servers Provider is the one who providers network for both general users including web service providers. All the on-line service rely on it. However,it is not hundred percent reliable by its users. By the physical view, routers, switches and cables can have the risk of be ruined. The

modems connecting from ISPs to end users can be out of work. Network link congestion, which is a basic cause for slow web environments, is a common issue for poor Internet service. The maintenance errors of base-station for a ISP can give a fatal effect on all its related users in lower levels.

- Users' Weak Hardware
  The hardware's problem can be the low configuration of itself. The non-mainstream computer configuration gives a user bad experience for surfing. In addition, the choice of different hubs or routers for users' connecting to the Internet has effect on the users' network experience. The thermal control of hardware is also essential for they can be too hot to run after keeping on working for a long-time.

- Software Problems
  Both the operating systems and the browsers for surfing the Internet can be varied for each user. Meanwhile, the plug-ins of browsers for specific applications like java platform, Adobe flash player requires to be installed or updated. Otherwise, certain website can not be running smoothly. And Some mistakes of network configuration is a common case letting users' down. Final, possible virus or worms that threatening network security can decrease users' satisfaction.

- Local network Contention
  The case that more than one devices sharing the only bandwidth which is inadequate for so many devices at the same time result in the competition of limited network resource. Such as a family network in a household, it has only one network connection from outside whereas connected by several inner side web devices.

- Infrastructure Failures
  The infrastructure is the connections of computers, all sorts of network devices in a network system. Some unexpected manual interventions are potential pitfalls for network infrastructure. Take a road work for example, which may cut off the cable embedded underground and collapse of related part of the network. Meanwhile, nature disasters e.g. thundering, flooding may damage network devices or any parts of the whole infrastructure that brings on the disconnection of the infrastructure. The damp or wet conditions can induce dangers for the physical network facilities.

2.2.3 Measurement methods of QoE

There exist many methods for measuring QoE in Telecommunication field, which can be references and rudiments of measurements of QoE for web services. Here introduced three of them.

Perceptual Evaluation of Speech Quality

Perceptual Evaluation of Speech Quality (abbreviate PESQ) is applicable not only to speech codecs but also to end-to-end measurements. The validation of PESQ included a number of experiments that specifically tested its performance across combinations of factors such as filtering, variable delay, coding distortions and channel errors.[24] PESQ has a complicated algorithm for averaging distortions over time.

Video Quality Measurement

Video Qualit Measurement is an objective measurement of video quality to make a bench mark of different proposed methods for objectively assessing video quality.
The grading scale is illustrated by DSCQS(Double Stimulus Continuous Quality Scale) method of ITU-R Rec. BT.500-10.

The scale used by the viewers goes from 0 to 100. In this study, the raw difference score were rescaled to a 0-1 scale. Scaling was performed for each subject individually across all data points. A scaled rating was calculated as follows:
scaled rating = (raw difference score – Min) (Max – Min)
where Max = largest raw difference score for that subject and Min = minimum raw difference score for that subject. [25]

Mean Opinion Score

Mean Opinion Score (abbreviation MOS) is defined in ITU-T Rec. P.10/G.100 in the following way:
The mean of opinion scores are of the values on a predefined scale that subjects assign to their opinion of the performance of the telephone transmission system used either for conversation or for listening to spoken material.[26]

In [23], methods and proce.dures for subjective determination of transmission quality is introduced. Mean Opinion Score(abbreviate MOS) is subjective compared with the two models above as it is based on what is perceived by users during the tests. The numerical scores of MOS, from 1 to 5, are for rating quality . The level goes advancing as the value increasing in sequential order. The experimenter allocates the following values to the scores:

* 5 – Perfect. Like face-to-face conversation or radio reception.
* 4 – Good. Imperfections can be perceived, but sound still clear. This is (supposedly) the range for cell phones.
* 3 – Fair.

* – Poor. Nearly impossible to communicate.
* – Bad.

MOS can simply be used to compare between VoIP services and providers. But more importantly, they are used to assess the work of codecs, which compress audio and video to save on bandwidth utilization but with a certain amount of drop in quality. MOS tests are then made for codecs in a certain environment.[23]

2.2.4 Related work with QoE

A Existed Framework for QoE

A framework for measuring Quality of Experience is implemented in [35]. Whenever a subject feels dissatisfied with a network applications in use, he/she can just perform on OneClick that aims for capturing users' perceptions.
PESQ and VQM, two objective quality assessment methods, are referred to demonstrate the validation of OneClick for evaluating the quality of the application.
Two applications, one for instant messaging applications, and the other for first-person shooter games, are cited for this framework's efficiency and effectiveness. It proves that the OneClick framework is application-independent and can be used to assess quality of experience in both interactive and non-interactive scenarios.
OneClick is available at *http://mmnet.iis.sinica.edu.tw/oneclick*. Anyone can contribute perceptions of the quality of the provided audio clips to form a composite perception for future QoE studies.

2.3 Firefox Add-ons

Mozilla Firefox is a free and open source web browser which runs on the current mainstream operating systems and many other platforms. As of February 2011, Firefox is the second most widely used browser, with approximately 30[3, 4, 5]

There are large amount of Firefox's Add-ons supporting and extending its functionality in two categories. One is plug-ins like Acrobat Reader, Java and Flash Player. Another one is extensions which provide a rich user experience. Extensions are different from plugins, which help the browser display specific content like playing multimedia files. Extensions are also different from search plugins, which plug additional search engines in the search bar.[6]

2.3.1 Prerequisites for implementing Firefox extension

Technologies in use

Firefox implements many web standards, including HTML, XML, XHTML, MathML, SVG 1.1 (partial), CSS (with extensions), ECMAScript (JavaScript), DOM, XSLT, XPath, and APNG (Animated PNG) images with alpha transparency.
[14]

In between them, the main technologies for building extensions are: XUL, CSS, JavaScript, and XPCOM. Different roles for each are illustrated by figure 2.1.[7]

Figure 2.1: role of each technology in Firefox

The minimum knowledge required for development of Firefox extension:

- Field 1: XML coding
- Field 2: CSS coding
- Field 3: Basic JavaScript syntax

*XML:*A text-based structural language Extensible Markup Language (XML) is a meta-language for expressing various kinds of data. It was specified in 1998 by W3C, the organization that sets standards for web-related technologies. It has a number of useful qualities: it is generic, extensible, and easy to validate as well-formed. Some examples of XML-based markup languages include XHTML, which is HTML redefined on an XML base; SVG, for expressing vector images; and MathML, for expressing mathematical formulas. XUL, which is used in Firefox, is also based on XML.[7]

CSS: A style language to alter the display of XML documents Like XML, Cascading Style Sheets (CSS) is a technical specification established by the W3C; it is a style-description language defining the display of data marked up in

XML and HTML. As shown in Listing 1, it has an extremely simple syntax. By separating the structure of the data, expressed through HTML or XML, and the display style, indicated by CSS, data can be reused better than it is when structural and stylistic markup are both embedded in HTML.[7]

JavaScript: The world's most misunderstood language
JavaScript is a prototype-based object-oriented scripting language that is dynamic, weakly typed and has first-class functions. It is also considered a functional programming language like Scheme and OCaml because it has closures and supports higher-order functions.
JavaScript is an implementation of the ECMAScript language standard and is primarily used in the form of client-side JavaScript, implemented as part of a web browser in order to provide enhanced user interfaces and dynamic websites. This enables programmatic access to computational objects within a host environment.[8] In[29], a complete JavaScript implementation is made up of the following three dinstinct parts:

* The Core(EMACScript)
* The Document Object Model(DOM)
* The Browser Object Model(BOM)

Introduction to Chrome

Chrome is the user interface parts of the application window that are outside of a window's content area. Taking Firefox web browser for example, tool-bars, menu bars, progress bars, and window title bars are all examples of elements that are typically part of the chrome.[9]
In the situation of Firefox's extensions, extensions overlap part of Firefox's chrome by modifying the user interface and adding specific functionality with JavaScript.
Chrome Providers

A supplier of chrome for a given window type (e.g., for the browser window) is called a chrome provider. There are three basic types of chrome providers:[9]

- content The main source file for a window description comes from the content provider, and it can be any file type viewable from within Mozilla. It will typically be a XUL file, since XUL is designed for describing the contents of windows and dialogs. The JavaScript files that define the user interface are also contained within the content packages, as well as most XBL binding files.
- locale Localizable applications keep all their localized information in locale providers. This allows translators to plug in a different chrome package to translate an application without altering the rest of the source code. The two main types of localizable files are DTD files and Java-style properties files.

- skin A skin provider is responsible for providing a complete set of files that describe the visual appearance of the chrome. Typically a skin provider will provide CSS files and images.

The providers work together to supply a complete set of chrome for a particular window, from the images on the tool bar buttons to the files that describe the text, contents and appearance of the window itself.

The Chrome Registry
The gecko runtime maintains a service known as the chrome registry that provides mappings from chrome package names to the physical location of chrome packages on disk. This chrome registry is configurable and persistent, and thus a user can install different chrome providers, and select a preferred skin and locale. In order to inform the chrome registry of the available chrome, the ”new-style” plaintext manifests.

Chrome Manifests
The plaintext chrome manifests are a simple line-based format:

- content package
  content packagename path/to/files
- locale package
  locale packagename localename path/to/files
- skin package
  skin packagename skinname path/to/files
- XUL overlay
  overlay chrome://file-to-overlay chrome://overlay-file
- Style overlay
  style chrome://file-to-style chrome://stylesheet-file

2.3.2 Setting up a Firefox extension's development environment

New development profile

Firstly, create a new profile for development. On Windows:

```
Start -> Run "firefox.exe -P"
```

Development preferences

For making debugging convenient, certain relevant development preferences need to be enabled to view more information about application activity. Since these operations are implemented in the developer profile, there is no need to consider the influence on browser performance when enabling these preferences. First thing for enable them is typing *about:config* in the location bar of Firefox.

Then, just follow the list's description, modifying the boolean value of each preference. In case that a preference mentioned below doesn't exist in default, still it can be created beforehand in the format of a new boolean entry.

- javascript.options.showInConsole = true. Then, the javascript errors will be report to Firefox's Error Console.
- nglayout.debug.disable_xul_cache = true. Disables the XUL cache so that changes to windows and dialogs do not require a restart.
- browser.dom.window.dump.enabled = true. Enables the use of the dump() statement to print to the standard console.
- javascript.options.strict = true. Enables strict JavaScript warnings in the Error Console.
- extensions.logging.enabled = true. This will send more detailed information about installation and update problems to the Error Console.
- nglayout.debug.disable_xul_fastload = true.
- dom.report_all_js_exceptions = true.

2.4 WAMPs(Apache, PHP and MySQL for Windows)

2.4.1 WAMP

WAMPs are packages of independently-created programs installed on computers that use a Microsoft Windows operating system. [10]
The acronym WAMP refers to a set of free (open source) applications, combined with Microsoft Windows, which are commonly used in Web server environments. The WAMP stack provides developers with the four key elements of a Web server: an operating system, database, Web server and Web scripting software. The combined usage of these programs is called a server stack. In this stack, Microsoft Windows is the operating system (OS), Apache is the Web server, MySQL handles the database components, while PHP is a scripting language that can manipulate information held in a database and generate web pages dynamically each time content is requested by a browser. Other programs may also be included in a package, such as phpMyAdmin which provides a graphical user interface for the MySQL database manager, or the alternative scripting languages Python or Perl.[11]

The equivalent installation on a Macintosh operating system is known as MAMP.
The equivalent installation on a Solaris operating system is known as SAMP.
The equivalent installation on a FreeBSD operating system is known as FAMP.

2.4.2 Apache web server

Apache webserver is one of the world's most popular webserver which is also open-source. Apache was originally designed for Unix users, and now also supports Windows and other platforms.

2.4.3 PHP programming language

PHP is a general-purpose scripting language originally designed for web development to produce dynamic web pages. For this purpose, PHP code is embedded into the HTML source document and interpreted by a web server with a PHP processor module, which generates the web page document. It also has evolved to include a command-line interface capability and can be used in standalone graphical applications. PHP can be deployed on most web servers and as a standalone interpreter, on almost every operating system and platform free of charge. PHP is installed on more than 20 million websites and 1 million web servers.[16]

2.4.4 MySQL

MySQL is a relational database management system that runs as a server providing multi-user access to a number of databases. It is named after developer Michael Widenius' daughter, My. The SQL phrase stands for Structured Query Language.
The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements. MySQL was owned and sponsored by a single for-profit firm, the Swedish company MySQL AB, now owned by Oracle Corporation.[15]

Chapter 3

Approach

This chapter discusses possible approaches of QoE of web services. In other words, the question to be solved in this chapter is ”how is the way of acquiring users' experience?” Three approaches are listed:

* On-line Questionnaire
* Website Rating System
* Web browser's Add-on

In between of these approaches, questionnaire is the common way for getting users' feedback, which is easy to implement. And website rating system is a relatively new way, which was originated from the practical voting and rating system and has many implemented examples on the Internet. The last one is based on web browser's add-on, which has many instances referring to many aspects and technologies for implementation supports.

3.1 On-line Questionnaire

The first approach thought about is by questionnaire, which has widely used in marketing research, health professionals and sociology. By given well-designed questions or multi-choice survey, it can acquire feedback from users to the web service provider about its web service.

Different modes of questionnaire administration are given below:[21]

* Face-to-face questionnaire administration, where an interviewer presents the items orally.
* Paper-and-pencil questionnaire administration, where the items are presented on paper.
* Computerized questionnaire administration, where the items are presented on the computer.

* Adaptive computerized questionnaire administration, where a selection of items is presented on the computer, and based on the answers on those items, the computer selects following items optimized for the tester's estimated ability or trait.

By the explanation of each mode, computerized questionnaire administration matches the QoE survey well. Since the users evaluate web service on-line, they always approach Internet easily, making computerized questionnaire practicable. A special channel, for example a textual button with explanation on it, can instruct users to attend the feedback activity by redirecting to a new web page containing questionnaire. After fulfilling all the answers and submission, the feedback will be transmitted into the back-end platform. An application can be executed for the storage of each incoming answer.
This approach has three advantages:

* Web service providers can get the answers they want by well-designed questions beforehand.
* It's easy to understand and operate by users.
* It's intuitive for on-line replying of questions.
* It's feasible for implementation.
  A web page containing all the questions, a program for fetching and storing answers, and a relevant database: these three parts consist the whole on-line questionnaire system.

However, drawbacks are also existed in this approach:

* The feedback process can be delayed artificially.
  A scenario could be: a user answer the questions half way, leave them for a while, and then pick up it. Whereas, what the back-end application accepts is a feedback a few while ago, but not the real time impression of the web service. At this moment, this user might have a different experience of the same web service.
* From the users' point of view, since the whole procedure of answering questions takes them a period of time, they might be inpatient in answering or just drop the process half way.
* For each web service vendor, a specific questionnaire system needs to be implemented and maintained. That means extra hardware, software and human resources are in demand.

3.2 Website Rating System

Rating is the evaluation or assessment of something, in terms of quality or quantity, or some combination of both.[12] There are diverse ratings on entertainments, educations, scientific researches and so on. One of the popular

format of ratings is rating site.
In the already existed voting websites, users can vote for their favourite design, most interested news and so on. Generally speaking, one voting website focuses on one of these topics for users vote or comment. For instance,a website *http://www.cssvote.com/* is a specific on-line voting system opening to anyone who can submit, bookmark, comment or vote for his/her favourite CSS/HTML designs. In addition, the voting result will be reviewed by all the users on the home page.

Until now, a new approach is inspired from this sort of rating website: a specific rating site can be set up for collecting QoE. by this site, any users can submit any website with or without good web service experience together with the submitted website's URL.
Technically, a third party will be responsible for the rating site, but not every web service provider. Both benefits and shortcomings of rating sites can be previewed:

* It is visualized since in the form of websites.
* It is achievable.
  The given example is implemented by an open source content management system software naming **Pligg**, which provides the function for creating rating websites for public distribution. Hence, it is imaged that a voting system for evaluating different web services by public users is possible to achieve.
* However, it is not time-effective.
  An experience for a previous browsed web page might be submitted to the rating site not that immediately. But experience can be varied in distinct time interval.
* And it is not ease to use.
  In a rating site, users have to type both the relevant website's URL and the rating result in order to evaluate a website's service quality.

Here draws a conclusion that rating site system is not qualified in time-effectiveness for web services' rating. Meanwhile, the operability from users' aspect is weak.

3.3 Browser's Add-ons

An improvement generated from the first two approaches is made in this new approach in the way of caring about users' operability. Feedbacks operation will conveniently locate in the submitted web page. More specifically, users don't need to redirect to any other web page or system for feedback, but just report on the current web page.
On the other side, when it comes to web services, the first thing to associate is the web browser. Besides, browser acts as a client in a HTTP request response

protocol which transmitting an HTTP request message from a client's submission to the server, the web service provider. So, spontaneously, web browser has the natural advantage for sending users' feedback.
Since, there are no web browsers supporting feedback process, it is hoped that a web browser can extend its function by adding an extra ability of dealing with users' feedback. As a result, web browser's add-ons, which can extend web browsers' functionalities, can qualify these characteristics above. Web browsers' add-ons will be the new approach.

Login issue

A traditional way of submitting feedback in a website ought to be logged in by users firstly. Whereas, in this approach, for the users' convenience, this login step is given up since it's too complicated to do.

Comments issue

Furthermore, the process of submission of users' experiences is discussed below.
First, here rises the idea of providing a text-box for users' typing of feelings of the current web pages. However, taking into consider of users patience, they are more likely to just give up the feedback process by closing the web pages, since it is troublesome to write down comments for users. For web service providers, they don't like to hear noisy voices from all sorts of people with each point of views. In addition, they may be not willing to hire new person to deal with opinions from users one by one. Anyhow, the avoidance of over heavy workload to an enterprise is important. And it is hard to distinguish whether the opinions are valuable or not and it takes time to prove their values. Finally, comments input is dropped.

Textual buttons

Meanwhile a new solution, which can simplify the users' submission operation, is requested. A possible way is to provide different levels of satisfaction degrees for users to choose from. Naturally, this multi-choices including different satisfying levels are an improved manner compared with the previous one. Also, the rating levels are well structured beforehand. That brings regulation for the unified judgement. Only the users need to do is to choose from the following options without writing any comments. Here below is an example of five levels of satisfaction degree illustrated by textual buttons. And the five levels are according to Mean Opinion Score(MOS).

Figure 3.1: Buttons for different rating levels

Image buttons

Furthermore, an intuitive interaction is critical and ought to be implemented. In other words, with appealing and vivid user interface might be a better choice than the monotonous textual way.
Chatting on the Internet is funny for people, they can send various emoticons besides typing words. Learning from that, the emoticons can be introduced to replace the textual buttons above.
Here is an adaptation with the emoticons describing the users' feelings. Each emoticon corresponds to a rating level which is in the same column.

Figure 3.2: Emoticons corresponding to textual expressions

The final decisions on a web browser's add-on will ease users and direct-view.

3.4 Comparison of The Three Approaches

If drawing an analogy that the implementation of QoE is like a play, it needs different roles to join in. There is no doubt that the Internet users is an essential role who express feeling by their "Experience". One prerequisite for drawing their attendance is **easy of use**. Also, an **intuitive** system for feedbacks is vital. At the same time, it is hoped that users' experience can make sense the web service providers. So feedbacks should be **time-effective**. Based on the three requested properties, a comparison of the three approaches are declared:

Table 3.1: Comparison of Three Approaches

Approach	intuitive	easy of use	time-effective
On-line questionnaire	good	medium	medium
Rating site	good	medium	medium
Browser's add-on	good	good	good

According to the table, a browser's add-on is the relatively best approach for feedback collection. Consequently, it is the way for researching the QoE of web services in this thesis.

Chapter 4

Design

After the discussion of approaches for QoE in web services, a web browser's add-on is selected as the test approach in this thesis. The design part is commenced from determining the overall skeleton of the QoE framework. Moreover, the specific consideration of each components will be discussed after that.

4.1 A Framework of The Software Approach

4.1.1 The original prototype

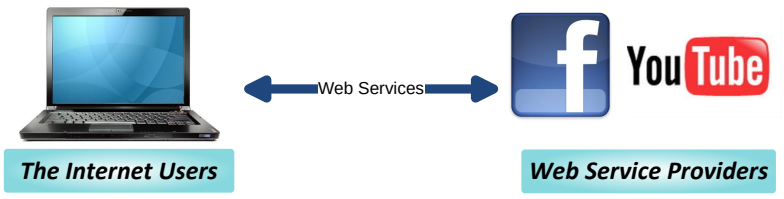

Figure 4.1: The Original Prototype of Implementation System

Figure 4.1 above shows the prototype of QoE framework. The end users who surf the Internet can communicate with the web service providers directly about their feelings when in need. Hence, service providers get to obtain each QoE by users' direct feedbacks.

Another similar way of illustration the relationship between users and service providers is given by figure 4.2. User a gives feedbacks to two web service provider A and B. Then, the two web service providers could adopt these feedbacks by adjusting themselves to possibly satisfy user a.
As for WSP B, it receives feedback from two users: a and d. Figure 4.2 illustrates a many to many relationship between users and WSPs that one end user can report to more than one web service providers if it is necessary, in the same

way feedbacks can be collected from all multifarious users.

This relationship is presented conceptually from two roles point of view:

* Supply feedbacks on users' side.
* Accept of feedbacks and followed adjustments on web service providers' side.

In other words, one end user report to different web service provider in separate time, e.g. user a in figure 4.2. Whereas, one web service provider, at an identical time, the feedback can be fetched from various of end users.
Feedback could make sense for some web service providers. Imagine that if

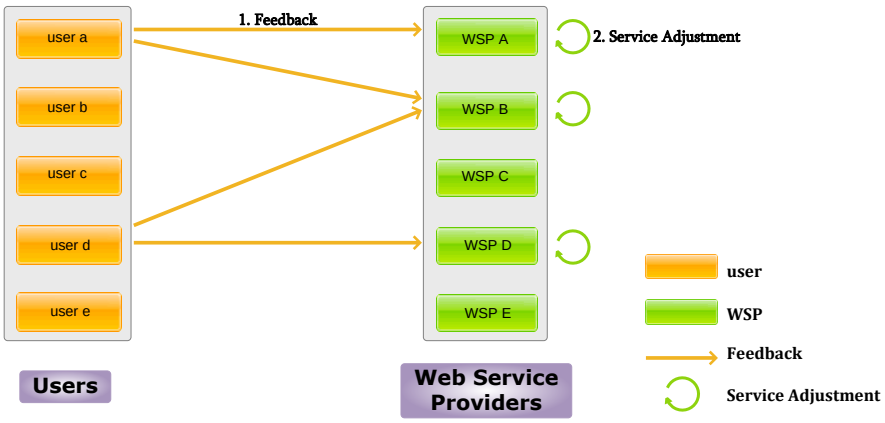

Figure 4.2: Conceptual Representation

there are abnormal number of users complaining about the same web service provider during a short time, the WSP is better to check whether it is a signal of bad performance or there exist some defects of itself.
Going deeper, the web service provider must monitor the incoming feedback from users from time to time so as to get the timely opinions. Then, the process of collecting feedbacks could be meaningful for service providers who always expect an better quality of their services.

Users submit feedbacks pro-actively, whereas service providers accept them passively. So service providers should keep deal with incoming feedbacks. Then, the system will be a real-time one. Mindful of the second step in figure 4.3 above, the real-time monitor and treatment will add the workload for web service providers. Each incoming feedback will be accepted, sorted, stored and processed. That will occupy part of the bandwidth of both sides. Besides, a specific server for users' feedback collecting could be considered then. Also, the physical storage and the possible equipments on this process should be considered by the web service providers.
Overall, extra resources will be consumed especially on each web service provider.

The conceptual representation is still too abstract to explain how the whole system works. So, it is necessary to start to go deeper based on the previous level.

4.1.2 A updated framework

On the other hand, it's foreseeable that probably not only one web service provider needs to do the same task above. Also, not all the service providers would like to know about users' feedback. In this sense, a new model on saving the resources for all these web service providers can be constructed and introduced.

Shifting from the distributed implementations on each of the web service providers to the centralized organization of the similar task can be a better choice. This means that on all the web service providers' side the common task of feedback process will be separated from all other applications and services, assigning it to another role to deal with. As a result, certain three party as a feedback process agency can be abstract from the discussion above.

Introduction of the certain party has the following benefits:

* It replaces all the web service providers which do the repeated task.
* The centralized storage of all the users' feedback avoid the web service providers' separate management of databases.
* On the network security's thinking, the defending of the probable threats by malicious attacks will be the third party's responsibility during the implementation.
* The physical resources are saved and workload are reduced for web service providers.

After the introduction of the new component, a new framework generates. Relationships between them can be expressed by the figure below:
As shown in Figure 4.3, the third party named Feedback Collector is the bridge for the communication between users and web service providers.

Furthermore, three components for implementation of QoE are specified and explained detailed in this way.

- Browser Feedback System The software tools for users' surfing the Internet are browsers, which must be the best choice of media for expressing users' feelings.
- Feedback Collector The third party, as the bridge between users and web service providers, focuses on the collection of feedback.

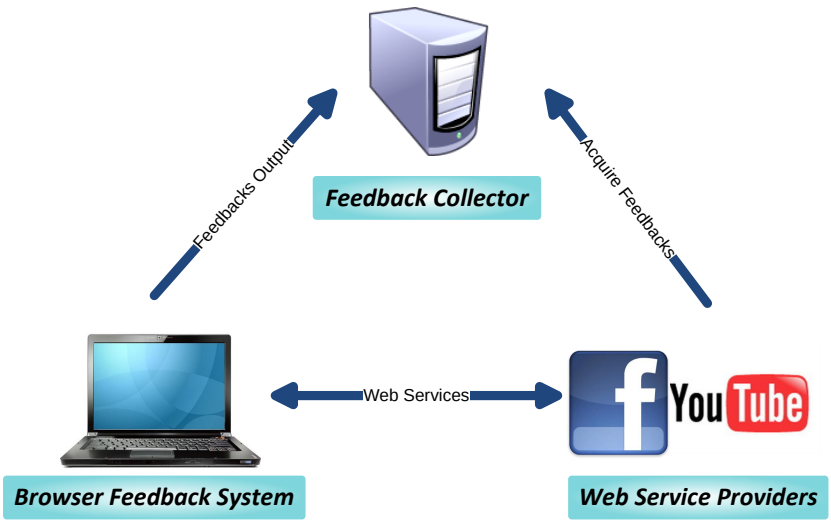

Figure 4.3: Three components of Implementation of QoE

- Web service Provider Web service provider is the one who get feedback of users from the Feedback Collector and make adjustment so as to enhance the quality of web service.

The figure below describe the final framework on technical consideration.

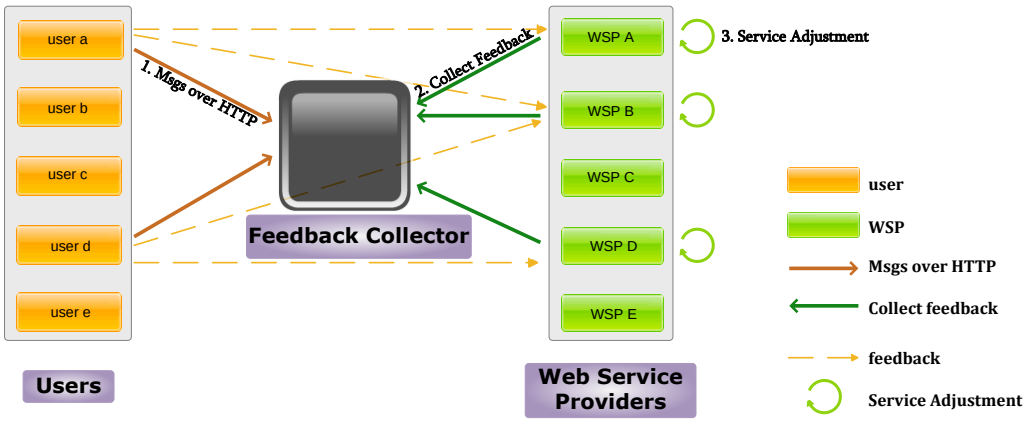

Figure 4.4: Technical Representation

In the technical level, the end users still feel like that they are talking to the web service providers by sending their feedback. Whereas, the feedbacks are sent in the form of HTTP messages to the feedback collector. As for web service providers, they can firstly decide whether to fetch these messages from the remote feedback collector or not. If yes, then the local service adjustments

policies could be made and implemented according to the feedback messages.

4.2 Browser Feedback System

As illustrated on figure 4.4, a http request from the web browser's user can be submitted to the intermediate server, Feedback Collector that can deal with the request and return a response to the user.

On one hand, the web browser itself doesn't support the feedback submission for users. On the other hand, add-ons of the mainstream web browsers can improve users' experience of a website in the way of providing multimedia or interactive content. Add-ons can be appended on the browser for extending its functionality.

The add-on is designed to supply an interactive user interface to users. Buttons in menu bar, status bar are a good choice which is intuitive and easy to operate for users. In addition, most of the current add-ons are implemented by this sort of buttons. Emoticons representing users' various feelings in figure 3.2 can place the normal textual buttons. However, the five-level rating mechanism, which is intuitive and easy to understand, has some drawbacks.

* People may feel hesitating and uncertain to estimate which level the current web site's network quality should belong to, since there are so many rating levels to choose from. On the other hand, from the practical viewpoint, the five levels make no much sense to the web service providers. Hence, the quality of user experience when using this feedback system falls down.

* From a psychological standpoint,the emoticons in blue color are contentious for blue always make people a bit down, so an alternative color need to be considered.

* The placement of five image buttons on the web browser will occupy too much space which result in the rising of impossibility of the implementation.

As a result, the following adaptation is given responding to these drawbacks one by one.

* The five-level rating mechanism is replaced by the good or bad rating mechanism. If the current web service is acceptable by the user, then the rating result is good this time. In contrast, if the user feels dissatisfying about it, then bad.

* The most common color for emoticons is yellow, so here it will replace the blue one. obviously, the yellow is neither good nor bad, which is a compromised choice, and can be accepted by the masses.

* The number of image buttons for add-on will be reduced to only one which is an icon representing the add-on. Moreover, a pop-up window is introduced for implementation of the centralized button. Once this image button is clicked, add-on will be activated and followed by a opened pop-up containing both the good and the bad choices.

A hypothesis of the pop-up window will be designed in this way.

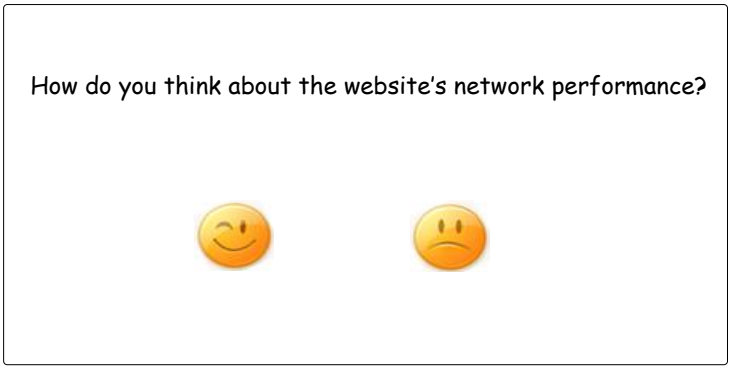

Figure 4.5: The hypothetical pop-up window of Add-on

The location of the image button of add-on is set in the status bar, where the user can click it conveniently compared with the one in the menu bar.

To satisfy the user more, the User Interface of this pop-up has a further update. When the user put the cursor onto the happy emoticon and prepare to press, it will zoom out automatically. Moreover, there appears a tip box which will show up when put the mouse on the happy emoticon by saying Ï'm happy.̈ And when the user press the happy emoticon, it will return to its original image size, making users feel they really press a button. In together, the tricks of changing the image's size can give a vivid experience by the animation effect when using the add-on.

Besides, since we only consider feedbacks about the quality of web service. The discontentment about the User Interface for example is not included in this paper. So the statement about the function of this add-on should be available easily by users. Right click of the add-on icon in the status bar is designed to inform users about what it is and how it functions.

Technically, the clicking the image button triggers the initiation of a pop-up window above the current web page. What's more, the user's operation on the pop-up window will send a feedback message from the user's browser to the feedback collector.
So, what can be included in this sort of message? Of course, the rating level

should be submitted. Since that the corresponded web page's URL is the unique identifier for locating this web page, its URL will be submitted. Obviously, how to let the feedback collector accept this http request and process it needs to be solved after this part.

4.3 Feedback Collector

Feedback Collector is an intermediate between Browser Feedback System and Web Service Provider. Accordingly, three tasks are cited:

* Deal with each separate http request from Browser Feedback System. Feedback Collector accepts each feedback message from users passively.
* Store every feedback message through browser add-on to a database. This database will be designed later.
* Provide the search function to all the Web Service Providers.

The figure 4.6 illustrates what are included inside of Feedback Collector and how it relates with the other two components.

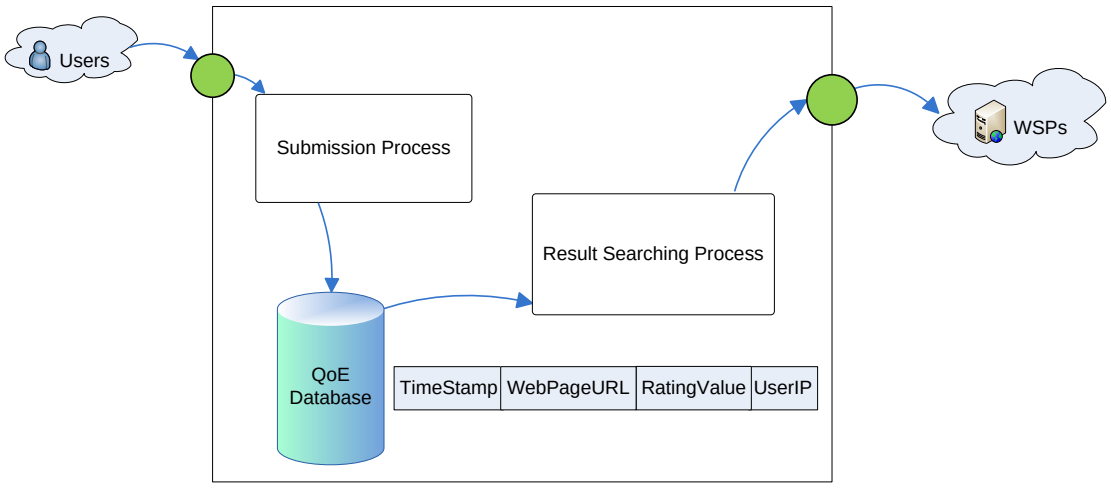

Figure 4.6: Feedback Collector Illustration

4.3.1 Storage of feedback data

For each feedback item in the database, these following fields are included.

* Date:
  The time stamp of each feedback request entering into the Feedback Collector.

* Reported web page's URL:
  The subject of current feedback is demonstrated by URL of itself.

* Rating value:
  Users choose different emoticons to report the feedbacks. Though, in the database numeric value will represent these feedbacks, with 0 being happy experience of users, 1 being unhappy.

* IP address:
  A further consideration is the IP address, since it can help to identify visitor's geographical location. So that, the statistical data of feedback by users' locations can assist WSPs to locate where are the web service problems.

4.3.2 Data search function

At the beginning, two approaches of providing these feedback data into each WSP.

* Pushing all of the messages to relevant WSPs, who accept them passively.

* Messages are available to every WSP, who will acquire them actively depending on their willing and demands.

The former approach doesn't consider WSP's requirements, since the users can report any web pages, while not all reported WSPs are really in need of these feedback. Meanwhile, it will consume large resource for keeping on pushing incoming feedbacks from users. However, the latter one makes up this drawback by WSP's pulling of messages only when required. Also it provides security guarantee than the former one.
By different searching requirements of each WSP, the responding searching result should be provided. For instance, searching result can be divided by a fixed time period, by a fixed IP segment, by certain keywords.

4.4 Web Service Providers

As seen from figure 4.3, there are two steps for web service provider.

* Collect Feedback

* Service Adjustment

The specific and detailed methods of the two steps are introduced below.

4.4.1 Methods for collecting feedback

For each web service provider, there exists a port on the Feedback Collector for providing associated feedback messages to the responding WSP. On the other hand, for each WSP, which is the way of acquiring feedbacks? It has an association with the searching function that Feedback Collector provides. Consequently, two corresponding ways are clarified here:

* pull based collection:
  Each web service provider fetches feedbacks on its own initiative.
* push based collection:
  Each web service provider passively receives feedbacks from Feedback collector.

4.4.2 Methods for web service adjustment of QoE

A poor service can be rooted from many parties. Here only adjustment based on web service providers are considered.

» Check if the power is off of all the network equipments, and if the cables are disconnected.

» Expand the network bandwidth especially there are extremely large number of users complain about the website happening at a fixed time period of each day.

» Set up more number of servers if a great many users suffer a bad experience co-instantaneously.
A web farm of multiple physical hosts grouped together will share the total load of the client requests

» Debugging of codes of the website.

» Examination of malicious attacks of website, of firewall of servers.

» Investigate the operation of website's data center.

» Inspection of inappropriate configuration or errors of servers.

Chapter 5

Implementation

Since the prototype and individual component have been well designed, this chapter will claim the real implementation of each component and the interactions between each other. Also the technical details will be revealed below.
Technology decisions are made for each component firstly. For instance, in browser feedback system, which web browser will be used for feedback system. Following implementation of extension involves more underlying aspects. Besides of deciding which server for dealing with feedbacks will run on feedback collector, the QoE database design is also specified. Moreover, the interaction with the other two components are described by two scripts. One for inserting new feedback, and another for function of feedback query. The last part will be implemented by an example scenario.

A conclusion of QoE system's architecture will be illustrated later.
A statement of how to distribute the extension and a test in small scope are given in the final.

5.1 Browser Feedback System

5.1.1 Decision making on web browser

Nowadays, IE, Chrome, Firefox and Opera are all mainstream browsers. The most flexible one for extending the browser's performance is Firefox. Up to now, Firefox has thousands of add-ons, which are developed by a great many developers around the world. Based on Firefox, installation of extensions allow the users to apply personalized browser.
There are many advantages of selecting Firefox.

* Firefox is popular. As of March 2011, Firefox is the second most widely used browser of the world.
* Open source atmosphere. Anyone can contribute to the development.
* Firefox is an easy, safe and extensible web browser.

* Firefox's extensions are lightweight to implement. The handling of feedback from users can be added as a new feature.
* Firefox's extensions are easy to publish and convenient to get by any online users. It has categories for all sorts of functional extension.
* Extension technologies for development are easy to grasp.

As a result, Browser Feedback System will be implemented by a Firefox extension.

5.1.2 The implementation process

The whole structure of the Firefox's extension

The extension is named "YourPageSucks". Here describes what directory structure and what files are needed in YouPageSucks.
Folder structure

```
test\_extension/
  chrome.manifest
  install.rdf
  chrome/
        content/
        test_extension.js
        browser.xul
        submit.xul
        about.xul
        icon.png
        smileyface.jpg
        sadface.jpg
        smileyfaceMouseOver.jpg
        sadfaceMouseOver.jpg
  skin/
        submit.css
```

Explanations of each file under the extension

chrome.manifest
chrome.manifest list all the chrome providers in this extension.

```
content   test_extension    chrome/content/
overlay   chrome://browser/content/browser.xul   chrome://\\test_extension/
    content/browser.xul
```

* The first line registers a content provider: it maps the contents of chrome://test_extension/content/ to the content folder.
* Line 2 registers an overlay for chrome://browser/content/browser.xul location, allowing developers to modify Firefox's main window UI from browser.xul file under test_extension/content/ directory. So that, browser.xul will modify Firefox's status bar by adding an icon of extension.

install.rdf
install.rdf file is an install manifest, used for providing information about YourPageSucks and registering this extension on Firefox. Its identifier, author, version and its compatible applications and more.

```
1     <em:id>admin@yourpagesucks.org</em:id>
2         <em:id>{3CC992FC-1F7C-475E-904F-2D4840C32B77}</em:id>
3         <em:version>1.0</em:version>
4         <em:type>2</em:type>
5         <em:name>YourPageSucks</em:name>
6         <em:description>Collecting web service feedback from general users</em
              :description>
7         <em:creator>Mei Yang</em:creator>
8         <em:homepageURL>http://www.mysite.org</em:homepageURL>
9         <em:iconURL>chrome://test_extension/content/icon.png</em:iconURL>
10        <em:aboutURL>chrome://test_extension/content/about.xul</em:aboutURL>

12    <!-- Firefox -->
13        <em:targetApplication>
14                <Description>
15                <em:id>{ec8030f7-c20a-464f-9b0e-13a3a9e97384}</em:id>
16                <em:minVersion>1.5</em:minVersion>
17                <em:maxVersion>4.*</em:maxVersion>
18                </Description>
19        </em:targetApplication>
```

* <**em:id**> is the unique identifier of this extension for distinguishing it with others.
* <**em:type**> represents the type of addon being installed. As an extension here, this number should be 2.
* <**em:targetApplication**>: since the extension is based on Firefox web browser, the target application is Firefox. Firefox's id, the minimum and maximum version supporting this extension are all described in **Description** block.

browser.xul
browser.xul is for overlapping *chrome://browser/content/browser.xul* by adding an extension icon on the status bar.

```
1  <script type="application/x-javascript" src="chrome://test_extension/content/
       test_extension.js"></script>
2    <statusbar id="status-bar">
3      <statusbarpanel insertafter="statusbar-progresspanel" context="
           test_extension-menu">
4        <image id="test_extension-statusicon" style="cursor:pointer;"
5            src="chrome://test_extension/content/icon.png"
6          tooltiptext="How do you evaluate the current web service?"
7          onclick="if(!event.button) { test_extension.run() }"/>
8      </statusbarpanel>
9      <menupopup id="test_extension-menu">
10       <menuitem label="Test Extension Homepage" oncommand="test_extension.
             goHome()"/>
11       <menuitem label="About" oncommand="test_extension.showAbout()" />
12       <menuseparator/>
13       <menuitem label="Run Test Extension" oncommand="test_extension.run()" />
14 ....
```

* The *script* line import the JavaScript file that the current .xul interacts with.

* The block of *statusbar* first adds the source file of the extension's icon. Also the *onclick* defines a trigger event that test_extension.run() function in test_extension.js will run when clicking this icon.

* The *menupopup* block initiates a menu of this extension when right click of its icon.

submit.xul
This file defines the elements and actions on the pop-up window.

```
....
<label id="reference" value="Evaluate the web page's web service quality here:
    " align="center"/>
....
        <hbox align="center" pack="center" resizable="false" width="80">
      <image src="chrome://test_extension/content/smileyface.jpg" name="
          jsbutton1" id="imageHappy"
          onmouseover="changeImageURL(this.id, 'chrome://test_extension/content
              /smileyfaceMouseOver.jpg')"
          onmouseout="changeImageURL(this.id, 'chrome://test_extension/content/
              smileyface.jpg')"
          onmousedown="changeImageURL(this.id, 'chrome://test_extension/content
              /smileyface.jpg')"
          onmouseup="window.opener.test_extension.sendRatingInformation(0);
              window.close();"
          tooltiptext="I'm Happy" />
    </hbox>
    <hbox align="center" pack="center" resizable="false" width="80">
      <image src="chrome://test_extension/content/sadface.jpg" name="jsbutton2
          " id="imageSad"
          onclick="window.opener.test_extension.sendRatingInformation(1);"
          onmouseover="changeImageURL(this.id, 'chrome://test_extension/
              content/sadfaceMouseOver.jpg')"
          onmouseout="changeImageURL(this.id, 'chrome://test_extension/content
              /sadface.jpg')"
          onmousedown="changeImageURL(this.id, 'chrome://test_extension/
              content/sadface.jpg')"
          onmouseup="window.opener.test_extension.sendRatingInformation(1);
              window.close()"
          tooltiptext="I'm Sad" />
    </hbox>
....
```

* The *label* illustrates the tip sentence for users about the function of this pop-up window.

* The block under the first *hbox* add an image button with smiley face on the pop-up window. Pointing the cursor above this image will show the explanation saying "I'm happy". Also different mouse events on the image button will change the image source file by invoking the changeImageURL() function in JavaScript. When clicking this image button, a good rating with value "0" together with current reported web page will be reported to the Feedback Collector by invoking another function sendRatingInformation().

* Another similar *hbox* followed by the previous one is for a bad experience submission with an unhappy face right of the smiley face.

test_extension.js
The behaviours of this test extension are implemented by *test_extension.js*. Here lists three main functions of it:

```
1  var test_extension = {
2    ....
3    function run() {
4          url = window.top.getBrowser().selectedBrowser.contentWindow.location.
               href.toString();
5          window.open("chrome://test_extension/content/submit.xul", "mywindow",
               "height=170, width=350, resizable=no, top=300, left = 500");
6    },
7    ....
8    function sendRatingInformation(rating)
9    {
10          var xmlHttp;
11          xmlHttp=new XMLHttpRequest();
12          xmlHttp.onreadystatechange=function(){}
13          xmlHttp.open("GET","http://ms4.vlab.iu.hio.no/qoe.php?rating="+rating
                +"&url="+url,true);
14       xmlHttp.send(null);
15     };
16     function changeImageURL(id,imageURL)
17     {
18           document.getElementById(id).src=imageURL;
19     };

21 };
```

* *run* function:
  It triggers a new pop-up window shown up on top of the current browser's window, with specific properties of height, width and so on. This new window is actually implemented by *submit.xul* file.

* *changeImageURL()*
  For mouse's different events, the URL of image button will be alerted.

* *sendRatingInformation()*
  AJAX(Asynchronous JavaScript and XML) is introduced to create interaction between the current reported web page and Feedback Collector. This web page's URL together with its rating value will be exchanged with the server in Feedback Collector without reloading the whole web page. Hence, users will feel that the web page is stable during the evaluating process.
  A new XMLHttpRequest is initialized first, which send the current feedback by opening a URL, for instance, *http://ms4.vlab.iu.hio.no/qoe.php?rating=0 &url=http://www.mysite.org*.

Finally, the execution of this extension is demonstrated by figure 5.1.

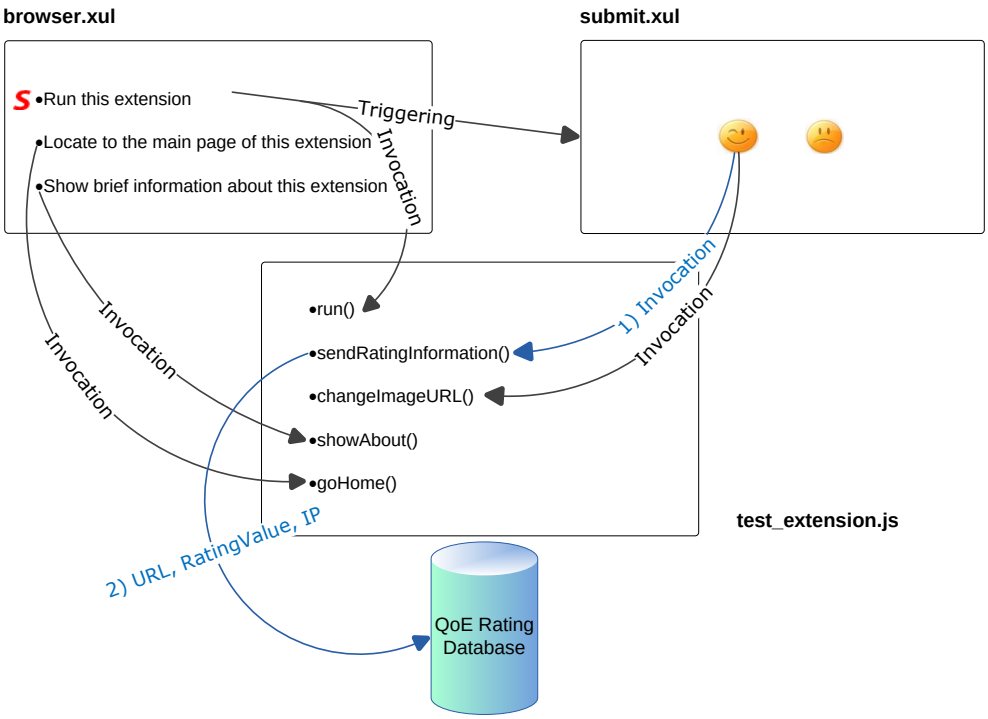

Figure 5.1: The programs invocation chart

5.1.3 The testing and demonstration of extension

First, the extension is installed by author locally for testing.
After installation, here opens any web page in Firefox web browser first, and then click YourPageSucks icon in the status bar. A window on top of the web page pops up as seen in the left screen-shot below. If pointing cursor above any of the emoticons, the corresponding one will be magnified as shown in the right screen-shot. Both Error Console and a development add-on of Firebug can be utilized for debugging of YourPageSucks. The latter one can be obtained by *http://getfirebug.com/*.

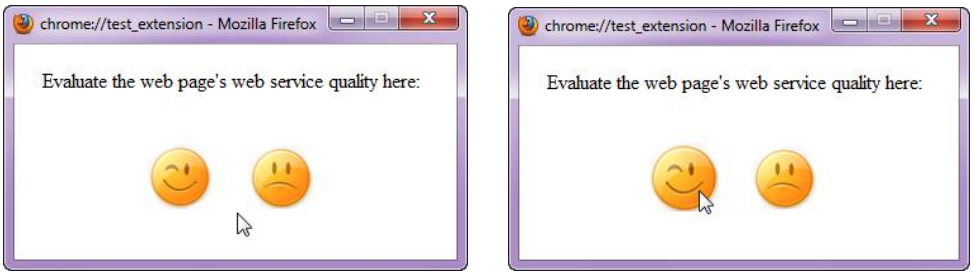

Figure 5.2: Screen-shots of extension

5.2 Feedback Collector

5.2.1 Incoming push-based message process

As illustrated in Figure 5.1, a http request is triggered once a user send a feedback message by clicking the relevant button on the extension. Meanwhile, the feedback messages will be pushed into the database of the Feedback Collector. The Apache web server is selected as the http server here, since it's popular, open source and easy to available. In together, the PHP scripting will assist to push data into a database.
qoe.php
Once the function of xmlHttp.open() in JavaScript is executed in feedback system, a connection to qoe_feedback database will be set up by qoe.php in feedback collector. And then insert the current feedback into the database with user's IP, rating value, and the relevant web page's URL.

```
1 ....
2   \$string = "insert into qoe_feedback (url,rating,ip) values ( '$url','
        $rating','$ip')";
3   mysql_query($string);
4 ....
```

The connected QoE database is designed in the form below:

Table 5.1: QoE Database

Index	Field	Type
1	date	DATETIME
2	IP	877
3	URL	varchar(256)
4	RatingValue	int(11)

5.2.2 Outgoing pull-based data searching

As known that feedback messages are open to anybody. However, how to exhibit the relevant messages to each WSP?
A graphical user interface can be one solution for implementing WSPs' searching function. It can looks nice, by a user-friendly interface. However, on second thoughts, the fixed look up items will limit the WSPs, since each of them has individual searching requirements. The GUI possibly will not solve each WSP's problems.

We can't guarantee a GUI for a system administrator of a website is useful, in stead, it is believe that creating an interface that allows programming in WSP's side is a better approach, since it allows system administrator to integrate the data to their monitoring tools, such as Munin.
Therefore, the choice was made to design an API(abbreviated application program interface) over http to conduct searches. That means, WSPs can utilize this API to pull the feedback from Feedback Collector and represent the result in their own ways. System administrators can create different tools depending on their needs and current monitoring tools in use.
qoe_search.php
This php script implements the API's searching function in two manners. Furthermore, a XML-formatted output is available for further speical use of each WSP.
Two manners of searching:

* query all the feedback items of a specific URL.
  Example 1: typing *http://ms4.vlab.iu.hio.no/qoe_search.php?query=mysite.org* in the location bar of web browser. It acquires all the feedback items about the URL of "mysite.org".

* query all the feedback items of a specific URL in a specific time period.
  Example 2: typing *http://128.39.73.238/qoe_search.php?query=mysite.org&lastm=5* It fetches the last five minutes' feedback about the URL of "mysite.org".

```
1 $link = mysql_connect("$dbhost", $dbuser,$dbpass);
2 ....
3     $string = "select * from qoe_feedback";
4             if ( $query and $lastm ){
5                 $string = $string . " where url like '%$query%' and date > NOW
                      () - INTERVAL $lastm MINUTE";
```

```
6               }
7               elseif( $query ){
8                   $string = $string . " where url like '%$query%'";
9               }

11              $result = mysql_query($string );
12              echo "<result>\n";

14              while ($row = mysql_fetch_assoc($result)) {
15                  echo "<feedbackitem>\n";
16                  echo "<url>" . $row['url'] . "</url>\n";
17                  echo "<rating>" . $row['rating'] . "</rating>\n";
18                  echo "<ip>" . $row['ip'] . "</ip>\n";
19                  echo "<date>" .$row['date'] . "</date>\n";
20                  echo "</feedbackitem>\n";
21              }
22              echo "</result>\n";
```

After the successful connection to qoe_feedback database, MySQL will be executed to get the required searching result. The final result is formed in a XML file. Here below gives the result example.

```
1  <result>
2          <feedbackitem>
3                  <url>http://www.mysite.org</url>
4                  <rating>1</rating>
5                  <ip>128.39.89.21</ip>
6                  <date>2011-05-06 12:12:15</date>
7          </feedbackitem>
8          <feedbackitem>
9                  <url>http://www.mysite.org/playvideo.php</url>
10                 <rating>1</rating>
11                 <ip>128.39.89.21</ip>
12                 <date>2011-05-06 12:12:16</date>
13         </feedbackitem>
14 </result>
```

As shown in the example, all feedback items are included in <result></result>. Each feedback item is placed in between a pair of <feedbackitem></feedbackitem>, where URL, rating value, IP, date of each feedback are all listed.

The overview of Feedback Collector's interactions are illustrated below:

5.3 Web Service Providers

For each individual, different tools can be developed for pulling data from Feedback Collector itself through the searching API.

5.3.1 Scenario one

www.mysite.org is the main page of the testing WSP. And it contains many pages underline for different contents. *www.mysite.org/playvideo.php* is a subpage of it for on-line video play. The whole website shares a group of servers. The assessment of servers is dynamic depending on the number of HTTP requests of each page. See in figure 5.2.

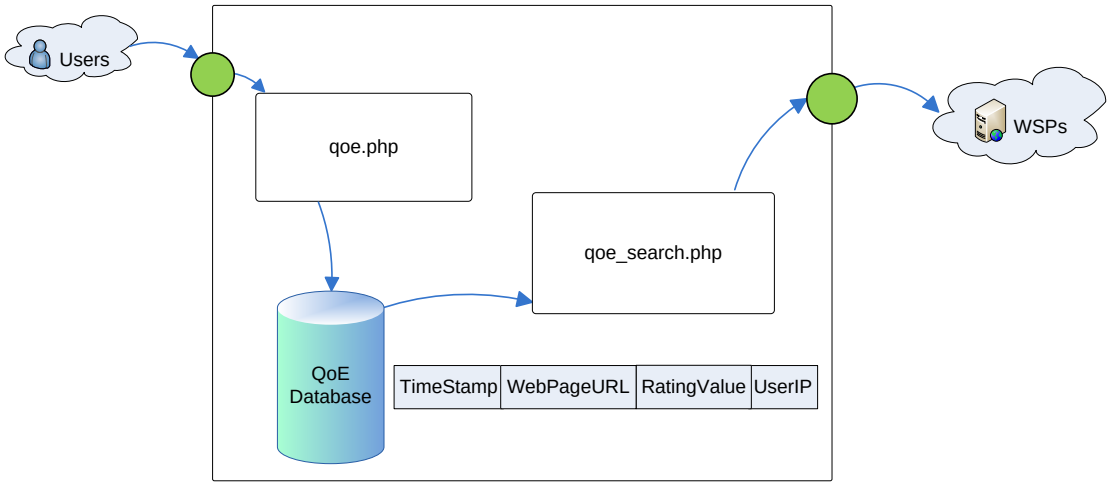

Figure 5.3: Illustration of interactions on Feedback Collector

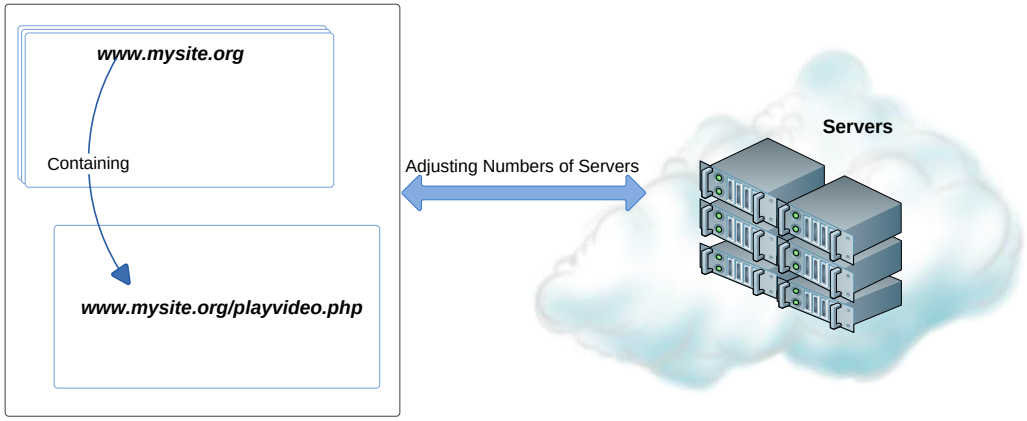

Figure 5.4: Illustration of webpages and servers of mysite.org

In the test, only feedbacks of both web pages will be discussed and represented in diagrams, which are updated every a few minutes. On the premise that the system administrator of this site uses Munin as the daily monitor tool of network performance. Munin produces nifty graphics that lets system administrator recognizes the possible problems in the current or upcoming time. For instance, the network resources trends can be graphed every five minutes. The system administrator determines to implement a Munin's plug-in as a tool for graphing the pulled data from the searching API. The plug-in is implemented by a perl script. Any feedback items with the URL of *http://www.mysite.org* or *http://www.mysite.org/playvideo.php* will satisfy the requiron the premise ofements of this WSP's searching.

```
1  # setting the query word of:
2  @QUERIES = ( "mysite.org", "playvideo.php" );

4  # The query interval is 5 minutes:
5  $INTERVAL = 5;
6  foreach my $query ( @QUERIES){
7      open(OUT,"wget -q -O - 'http://128.39.73.238/qoe\_search.php?query=$query&
           lastm=$INTERVAL' |");
8      my $counter = 0;
9      while ( my $line = <OUT> ){
10                 if ( $line =~ /<rating>1/ ){
11                 $counter++;
12                 }
13     }
14     \$query =~ s/\./_/;
15 }
```

Since Munin updates data every five minutes, the time interval of this searching plug-in is set to be the same. When cron runs Munin, the perl script is also running counting the number of new feedback items in five minutes. At the same time, the graphs are drawn and updated. And the feedback graphs can be categorised by day, month, year.
Also, it has statistical data about the current, the minimum, the maximum number of unhappy feedback.
Since *mysite.org* covers *mysite.org/playvideo.php*, the number of the former one is always greater than or equal to the latter one.
By the daily or monthly graphs generated from Munin-plugin, they can conduct service providers' internal adjustments. A typical example is tuning the servers assessment. More servers can be assigned to provide applications on *http://www.mysite.org/playvideo.php*.

5.4 Analysis of the System Architecture

The QoE system has three layers which are User Interface Layer(UI), Business Logic Layer (BLL) and Data Access Layer(DAL) as illustrated in Figure 14.

- UI Layer: It presents web browser's windows. When YourpageSucks is installed, it will overlay the window by XUL files under the BLL.

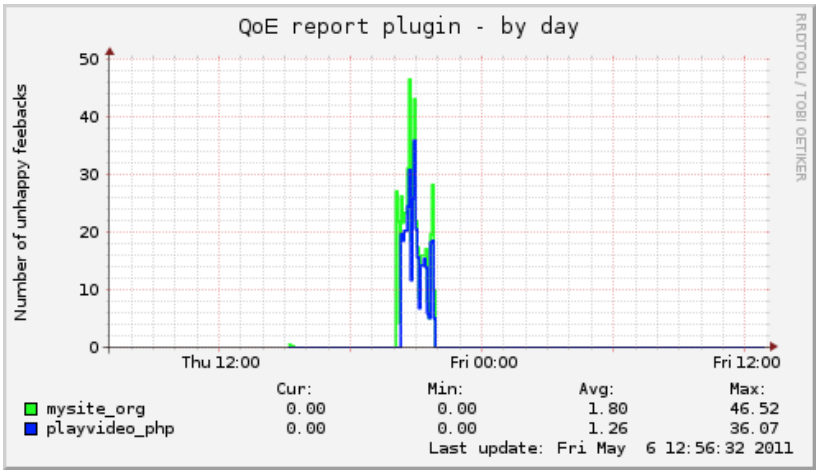

Figure 5.5: Numbers of unhappy feedbacks by day

- Business Logic Layer: It interacts with both the upper and lower layers. Three interaction events are claimed in this system.
  - XUL files in YourPageSucks overlay the browser window.
  - Execution of JavaScript will set up requestions of transmitting feedbacks to Feedback Collector.
  - Searching operation from Munin plug-in is dealt by underlying searching API.
- Data Access Layer: This layer manipulate database directly by SQL, which matters operations of reading and writing of data.
  - Writes: Each HTTP request triggered by JavaScript will be written into
  - Reads: When WSP searches for feedbacks, searching API will set up a connection to database and query the results.

5.5 Distribution of YourPageSucks

Preparations beforehand:

- Compress all the directories and files under **test_extension**, and then rename the zip file into a YourPageSucks.xpi file.[13]
- Register *www.yourpagesucks.org/net/com* as the main page of the QoE system in the future.

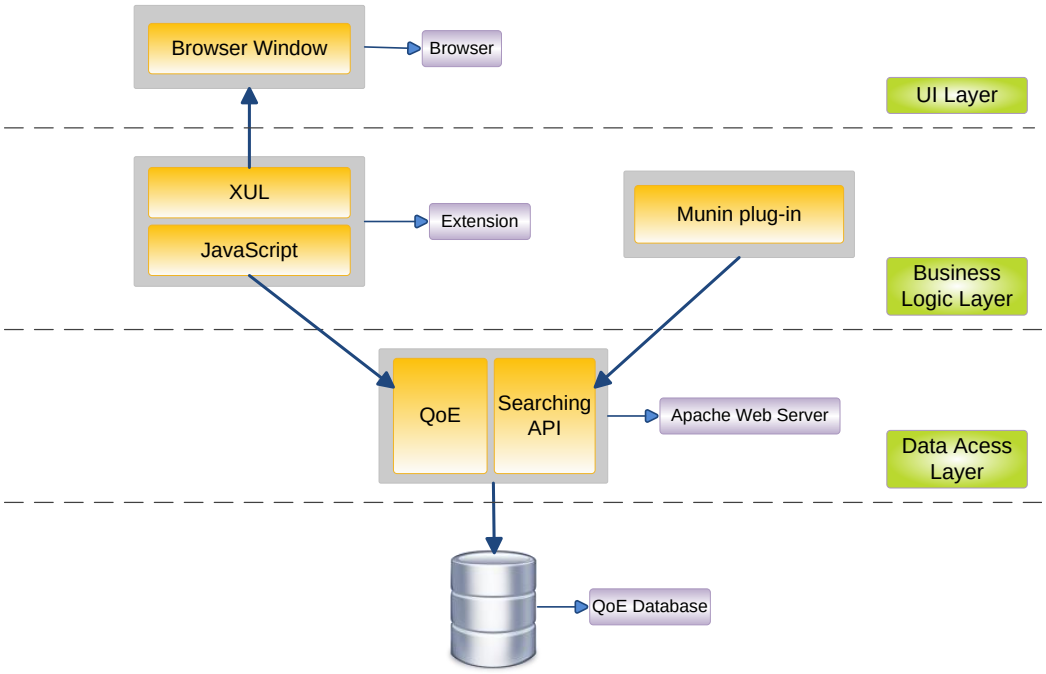

Figure 5.6: Three Layers Architecture

Then, it is ready. Go to the Firefox Add-ons Distribution main page *https://addons.mozilla.org/en-US/developers/*, and click Submit Your add-on button, upload the YourPageSucks.xpi file. After a few steps of submission, the whole distribution process has been done. Until now, YourPageSucks will have a public downloading web page in Firefox with detailed information.

As for general users, just by typing *https://addons.mozilla.org/en-US/firefox/addon/yourpagesucks/* in Firefox's location bar, YourPageSucks will be available.
Clicking "Add to Firefox" will install the extension to the current Firefox web browser. After restarting of Firefox, the extension can be executed for each user.

5.6 Test with Human Participants

Here introduces a testing scenario of the whole system. A testing group is specified with ten testers. First, they are asked to whether have Firefox web browser. After making sure Firefox installed, they will install YourPageSucks by *https://addons.mozilla.org/en-US/firefox/addon/yourpagesucks/*. After installation and fully use of YourPageSucks, they are asked to answer the following questions.

- What is the version of your Firefox web browser?

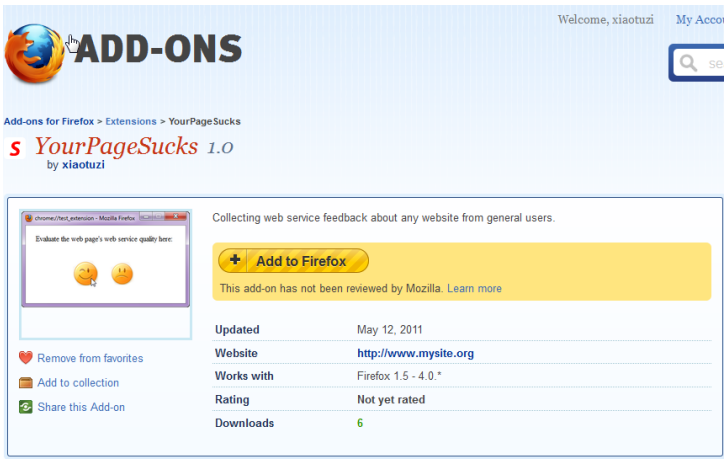

Figure 5.7: Screen-shot of YourPageSucks main page

- Would you be interested in installing YourPageSucks?
- What do you think about the user interface of the pop-up window revealed by clicking the extension icon?
- Do you believe the current number of rating levels(happy and unhappy) are enough or redundant?
- Are there any improvements that could benefit this extension?

The answers of these questions above are stated:

- All of the ten have Firefox 4 installed.
- Four users show their interests in installing this extension, and one user replied medium interest. Another four users are not sure about this question, and one have no interest.
- Five users satisfied the current UI design of the extension, and two considered it not too bad, and the rest dropped this question.
- Five users believe more levels can be added into the rating system. Two said only one is enough. Three agreed on the current design.
- Three of suggests from testers: one consider about the security of users' feedbacks; and another inquire the benefits for users from the extension; the last one had a problem of showing the whole textual line in the pop-up window, which results from the differentiation of screen resolutions of each computer.
- In addition, One user misunderstood the extension's purpose to be the feedback about the website's content whether it is interesting or not, or page layout whether it is clear and pretty or not. Another user felt confused about the extension's function at the very beginning of the test.

Chapter 6

Discussion

This chapter concludes the QoE research in this project. Also it gives possible improvements on each component, testing work, security issues and so on.

6.1 Security Issues

Since security is always an ingredient of a robust system, it is discussed in this section.

6.1.1 Rethinking about pop-up window

On network security consideration, the pop-up window is always controversial since it can be misunderstood as an embedded Ads or even a malicious web page.
Another solution is just setting up two buttons on the status bar representing happy and unhappy experience of users. Pop-up windows will not occupy too much space of the status bar, but it needs the users' trust.

6.1.2 Possible authentication of pulling feedback data

The feedback messages are collected from users following their willings, but not cheating from each WSPs, so these data doesn't seem to be that confidential. As a result, the feedback database is open to anybody in this implementation. On one hand, it has benefit that the Feedback Collector doesn't need to spend extra effort on set up the process of authentication. On the other hand, it saves time for each feedback seeker.
However, opening the feedback data to anyone always make the relevant WSPs uneasy and apprehensive. Hence, it is believed that asking for user name and password for each feedback seeker can relieve the uneasiness. It will also avoid the possible attack of feedback database in Feedback Collector by adding a protection mechanism on database.

6.2 Firefox Extension Difficulties

Difficulties of implementing Firefox extension are a bit time-consuming during the implementation.

6.2.1 Reference materials limitation

The resources of development of Firefox extension are not available systemically. Almost all the recipes of developing Firefox extension are from Internet in the form of web pages. Without knowing which recipe is right, developers have to try one by one or combine different recipes together to make a new one. In addition, because of the version problems of Firefox, some recipes can possibly not be compatible of new version. Even though the inconvenience of lacking systemic references, this kind of open source system development is a great experience with a great gain:

- Though references on the Internet can possibly not that trustful, the developers can still filter them. The official websites of each application always cover a mass of useful references.
- Predict the time consumption before try any development examples, so as to organize the whole schedule.

6.2.2 Programming difficulties

Though having the basic knowledge about JavaScript, it is still a bit tough for its application in Firefox's extension. The debugging tools during the development of extension are helpful but not enough. It is time-consuming for discovering bugs. Perhaps take advantage of some Firefox Add-on SDK could help the development of extension in the next time.

6.3 Future work on Browser Feedback System

The future work on Feedback System concerns the scale of feedbacks and the number of attendants. Similar feedback systems based on other browsers can increase the feedbacks' scale. Meanwhile, appending the interesting features or importing the reward mechanism will encourage the users' attendance.

6.3.1 Extensions for various browsers

In this research, feedback system is implemented only by Firefox's extension. However, as known that different users may use separate browser, it implies that the collected feedback data for the moment is not comprehensive for each WSP in scale. Hence, in the future, the Browser Feedback System can be implemented by all the main stream browsers. The more users' feedbacks, the more significant feedbacks will be for evaluation of web service.

6.3.2 Add interests into the feedback system

Suppose more users involves in using this extension, the Feedback Collector will be able to provide more feedbacks to WSPs. Hence, it is believed that a more attractive extension will attract more users so as to enhance the accuracy of feedbacks.
Something funny could be appended into the current feedback system. Like tell users daily constellation, fortune or their geographical information each time they use the system.
Or some more valuable information, for instance, feedback system could provide users some websites which have analogous functions for the reported websites. Since a great many websites provide on-line video services, if one of them are reported by a user, no matter good or bad feedback, the user would receive other similar on-line video websites. Comment on this adaptation of feedback system:

- Multiple functions for users might be able to attract their attendance. And it considers the users' experience of the feedback process.
- However, probably some of them dislike these functions, since each individual's expectation, fancy is not all the same. Meanwhile, extra implementation of these functions will be done on feedback system, with the support from back-end platform of Feedback Collector.

6.3.3 Integration with other applications

Multiple feedback system

Besides the web service feedback, various of feedback can be reported as well. For instance, the opinions about the website's content, the layout of the web page, can also be evaluated by general users.

6.3.4 Reward mechanism

In real business, the reward mechanism is for motivating users' attendance. Hence, the one who submit a great many feedbacks could be rewarded with a fixed amount of money. In addition, the reward levels could respond to the number of feedbacks per month or per week. This approach perhaps can draw users' attraction to attend the feedback, however, the operation of reward mechanism will consume extra human resources to implement and financial expense for paying awardees.

6.4 Probable future work on Feedback Collector and Web Service Providers

Plenty of time was spent on feedback system, which has been discussed with others. As a result, there is less time for the remaining two components. Sup-

posing to do it again, it will still take most time on browser feedback system as this time. So, it's believed that the decision making is right here.
However, there are some unsolved challenges at the two remaining components. These challenges are mostly related to the implementation but not the design. Consequently, future work could focus on developing a richer prototype, so that more testing and data gathering can be possible. Here gives examples of possible future work on Feedback Collector.

6.4.1 Feedback Collector

The searching API can extend its functionalities. For instance, the IP addresses of users can be fully utilized. Another keyword about IP can be included. Therefore, the WSPs can identify users' geographical location, which can help to locate where the web service possibly has problems massively.
Another aspect is for preventing possible several times of feedback on the same website in a short time period. It is understood that a quick temper can repeatedly click the same image button of this extension. So, the Feedback System doesn't prevent users to do so. However, in Feedback Collector, each new incoming feedback item will be examined by the user's IP to avoid repeated submission.
Corresponding to add some extra interesting or useful sub-functions on Feedback System, the Feedback Collector would provide back-end support. Continue to take *providing analogous websites* for example. The feedback Collector could categorise different websites by their contents and functions. As a result, each feedback submission will invoke a searching process of other similar websites and then display them to the relevant user.

6.5 Future testing work of the system

Since YourPageSucks is published on Firefox's official website, many familiar friends are asked to download the extension and try to use it. During the local test, some bugs are discovered and suggestions are raised. What comes next is further test in a wider scale. Here are two scenarios of the further testing.

- Testers are all the students in Oslo University College.
  In the help of system administrator of Oslo University College, the extension can be installed on all the public machines of college. Meanwhile, by the mail system inside of the college, the introduction and spread of YourPageSucks could be sent to every student. Any student interested in trying this extension will be the tester.

- Since having published it on Mozilla Firefox's Add-ons website, it is open to everyone.
  It means abundant feedback data for further analytical research on QoE can be received on the Feedback Collector. For instance, the popularity of the extension can be illustrated by the number of both downloads and

the feedback items. Possible opinions about the extension itself could be attained by individual users.

6.6 Controversy between generality/universality and specialization

6.6.1 A specific Feedback Collector in this research

This research is a general QoE framework for web service, not specific for individual Web Service Provider. That means all web server providers share the same Feedback Collector with the same search API. It has benefits in two aspects.

- Only a single implementation on Feedback Collector can help all WSPs to do a same task, saving time and money, other resources.

In spite of that, drawbacks are also existed.

- The single searching API can't always meet each WSP's individual requirement. For example, a WSP has a new requirement about users' IP addresses, which is not supported by the current searching API.

6.6.2 Both general and specific characteristics of Browser Feedback System

- The Feedback System is a general component for users' evaluation of web service. As for users, by installing only one extension, all the web pages can be submitted.
- On another point of view, the Feedback System is specific for web service evaluation only. The single feedback function may loose some people's attendance. Also, they can only evaluate by two options but can't comment on the reported website.

6.6.3 Two cases of specific occasions of the whole system

This research is standing on the general point of view for feedback collection from users and transmission to WSPs. However, specific cases can be an alternative implementation of the whole system.

Military Application

In a certain area of local area network, web services of this LAN are only open to users inside of this area. Take a military organization for example. If any employee of this organization can not access the its website, he/she can give a feedback to the feedback collector, since the failure of access could be fatal in military affairs. The web service provider may could pull the feedback in each

time interval. Or feedback collector could pushing the real-time feedbacks to this WSP, so that the WSP could make adjustments or self-check without too much delay.

- A mandatory installation and usage of this extension could be requested for each employee, to promise the instant on-line communication of martial stuff.
  In general occasions, it is not practicable for enforceable requirements for those unknown users.

- A promise of WSP's adjustment after receiving feedbacks is possible, since known that this WSP is eager to provide high quality of web service beforehand.
  Again, in general applications, it is impossible to guarantee each relevant WSP's attendance and to request it to make adjustments.

- In this case the push based feedback transmission, which saving the resource consumption during this process, replies on two conditions. The first one is that the number of users are limited in a fixed scale. Meanwhile it is sure that the WSP would like to fetch the instant feedbacks.
  However, in normal cases, because of the unpredictability of users' number, and the uncertainty of whether each WSP require feedbacks or not, the push based transmission doesn't make sense for general application.

- The push based feedbacks transmission ensures the timely adjustment on WSP's side. A possible improvement can be perceived on the users' side in the form of accessibility of network for instance. Whereas, there is no guarantee about timely adjustments in general occasions.

University Application

Still in a LAN but in a university, all the staff and students of the university browse its website, which always includes thousands of web pages. Typically, a slow unloading of a web page occasion users' vexation. Since the services and resources from the school's website are crucial for every user, they are willing to give a feedback when suffering a bad experience.
Hence, the system administrators of the school's website can pull these feedbacks for optimizing the network resources. On the whole, both the users and system administrators have the motivation of working on the system. And the feedbacks can truly help for system adjustments.

6.7 Possible variations in Feedback interface

6.7.1 The scale of rating levels

As seen that in this research, two levels of rating are given to choose from for users. It is easy to implement. Still, there are some alternative ways.

* Only one rating level for reporting bad experience of users.
  It is convenient for users
* More than two rating levels for choosing.

A voting for the rating levels could be an alternative way if doing it again. Based on statistics bases, the decision of rating levels will be more rational. However, it might be impossible to get a sufficient number to make it convincing.

6.7.2 User Interface

It is difficult to distinguish which choice of UI design is better. Different users always have separate opinions on the emoticons, the extension icon, and all other user interface elements. So, it is expected that the feedback about the interface can be obtained from the public users of this extension.

6.8 The reliability of users' feedbacks

It is hoped that all the feedbacks from users are valid and reliable. However, some feedbacks might mislead WSPs since they are not trustful. Two categories of this kind of feedbacks are given: the first one is out of users' control that they mistakenly blame WSPs for non-WSPs problems. And users always have no aware about their mistakes. The second is with users' awareness.

6.8.1 Noise interference in feedbacks

There can be various causes for a bad user experience as discussed before. However, the user will report a bad experience no matter what it results from. So, all the feedbacks which do not result from WSPs are called noise. If during a long time of feedback collection with plenteous data, it is predicted that the noise level is constant in a range by boosting or decreasing from time to time. Whereas, a sharp increase might imply an unusual incident which can be an evidence for WSP's adjustment.

6.8.2 Unfair ratings in feedbacks

Unfair ratings refer to the low reputation ratings based on the personal characteristics and intentions. The unfair ratings can be classified into two categories:[19]

- Individual unreliable ratings: an individual user provides unfairly high or low ratings subjectively. This type of ratings may result from users' personality/habit, irresponsibleness, and randomness.
- Collaborative unreliable ratings: a group of users provide unfairly high or low ratings to boost or downgrade the overall ratings of an object. This type of rating may due to the strategic manipulation from the owner of the object.

According to statistical bases, if more and more users submit feedbacks, individual unreliable ratings can be unapparent and controlled under a reasonable scope. So, they can be ignored ideally. However, collaborative unreliable ratings are more complicated.

6.8.3 future work on reliability of feedbacks

Definition Part:

- The total number of feedbacks N.
- Number of noise interference feedbacks, Nn.
- Number of individual unreliable ratings, Ni,
- Number of collaborative unreliable ratings, Nc.
- Number of reliable feedbacks Nr.

The impact value of unreliability factors can be calculated by the equation below:

$$IMPACT = \frac{(N_n + N_i + N_c)}{N} \tag{6.1}$$

If the total number of feedbacks approaches to infinite, the impact value would close to zero. As a result, for removing these unreliable factors, numerous feedbacks are needed for lowering the impact of unreliable factors.

6.9 QoE Research

As QoE is a new concept without much real implementation, it is open and free for any researchers who can create a new topic in this field. Researchers with different background always stand on individual angle for QoE. And it is exciting to starting the journey of QoE even without any specific knowledge of it beforehand.
However, the freedom is also a challenge since there is no matured scientific definition or researching boundary of QoE. In this project, it is always not easy to make a decision for each component's design and implementation. These decisions are not absolutely correct or wrong, better or worse. Looking back, I think this project and experiences can help future researchers, finding and refining QoE research, which is still a very open subject. One merit is that this research breaks two traditions in general.
One is that QoS is always an approach for measuring and evaluating web services quality. And another one is that most of the QoE and QoS researches focus on real-time service, like telephony and streaming video, however, here the research of QoE is about any general web service.

6.9.1 QoE and QoS comparison

Both QoE and QoS are studies of quality metrics. Researchers involved in quality study, should have a clear definition of how to measure and evaluate it and how it is defined and characterized.[28]

The implementation of QoE in this research is more close to an qualitative analysis. There is no computational model for QoE's further analysis. Users' feedbacks are the subject for measurement of web services' quality.
However, QoS studies lay particular emphasis on theoretical researches and accurate models in the form of QoS standards and mechanisms. Attributes of QoS's evaluation are categorized and varied. For instance, in[31], reliability, security, cost and performance are criteria to characterize QoS.
On this point, our QoE study has a new view point and an extension compared with the traditional quality metrics study.

As for web services, since the traditional way of web services' evaluation is more based on QoS, QoE could be introduced into the evaluation model in the future. The combination of both will be more comprehensive.

Chapter 7

Conclusion

The research work in this project has been developing a prototype of QoE for achieving satisfaction of user experience and provide it to service providers. As QoE is a new phenomenon relatively, its research is free and open. Therefore, a specific field of web services is confirmed as the beginning point of our QoE journey in this thesis. It is believed that users' perspectives on a web service can reflect its quality to a certain extent.

One highlight of this thesis is developing a Firefox's extension to be Feedback System. The implemented extension on Firefox is light-weighted for installation and use. So users' experience about web services will be transmitted by the extension firstly. Furthermore, in the form of extension, the feedback data can be stored into database more convenient by executing relevant JavaScript. What deserves special mention is that a third party between users and service providers is raised in this prototype. It has interaction with extension that accepting all the http requests from each feedback submission. In addition, it supplies service providers with a searching API for feedbacks collection.
It is unpredictable how each WSP would utilize these feedbacks, which stand for QoE of the web service. However, a practicable example in the form of Munin plug-in is introduced. Accordingly, QoE is characterized in the form of diagrams, which illustrate numbers of happy or unhappy users in different time interval. Of course, all sorts of ways of utilization of feedback data could be implemented depending on individual WSP's requirements.
Until now, the framework can brings people a straightforward impression of QoE and accumulate the practical studies of this term. And future work on statistical and analytical aspects could be actualized by larger scale implementation of this project.

From another point of view, this project has sought an alternative approach for evaluating web services at the same time. For one thing, it is distinct from a traditional way of QoS-based evaluation. For another, it is an innovation of traditional QoS-based evaluation. As a result, QoE and QoS could complement each other for web service administration and resources management in the future.

Appendix A

Source Codes

A.1 Browser Feedback System

A.1.1 chrome.manifest

```
content test_extension chrome/content/
overlay chrome://browser/content/browser.xul chrome://test_extension/content/
    browser.xul
locale test_extension en-US locale/en-US/
```

A.1.2 install.rdf

```
<?xml version="1.0"?>
<RDF xmlns="http://www.w3.org/1999/02/22-rdf-syntax-ns\#" xmlns:em="http://www
    .mozilla.org/2004/em-rdf\#">
  <Description about="urn:mozilla:install-manifest">
        <em:id>{3CC992FC-1F7C-475E-904F-2D4840C32B77}</em:id>
        <em:version>1.1</em:version>
        <em:type>2</em:type>


    <!-- Firefox -->
        <em:targetApplication>
                <Description>
                <em:id>{ec8030f7-c20a-464f-9b0e-13a3a9e97384}</em:id>
                <em:minVersion>1.5</em:minVersion>
                <em:maxVersion>4.0.*</em:maxVersion>
                </Description>
        </em:targetApplication>

        <em:name>YourPageSucks</em:name>
        <em:description>Collecting web service feedback from general users</em
            :description>
        <em:creator>Mei Yang</em:creator>

        <em:homepageURL>http://www.mysite.org</em:homepageURL>
        <em:iconURL>chrome://test_extension/content/icon.png</em:iconURL>
        <em:aboutURL>chrome://test_extension/content/about.xul</em:aboutURL>


  </Description>
</RDF>
```

A.1.3 browser.xul

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet href="chrome://test_extension/skin/submit.css" type="text/css"?>

<overlay xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul"
      xmlns:html="http://www.w3.org/1999/xhtml" id="test_extension-overlay">
  <!--Invoke JavaScript file: test_extension.js-->
  <script type="application/x-javascript" src="chrome://test_extension/content
      /test_extension.js"></script>

  <statusbar id="status-bar">
    <statusbarpanel insertafter="statusbar-progresspanel" context="
        test_extension-menu">
      <image id="test_extension-statusicon" style="cursor:pointer;"
          src="chrome://test_extension/content/icon.png"
        tooltiptext="How do you like the webpage?"
        onclick="if(!event.button) { test_extension.run() }"/>
    </statusbarpanel>

    <menupopup id="test_extension-menu">
      <menuitem label="Test Extension Homepage" oncommand="test_extension.
          goHome()"/>
      <menuitem label="About" oncommand="test_extension.showAbout()" />
      <menuseparator/>
      <menuitem label="Run Test Extension" oncommand="test_extension.run()" />
    </menupopup>
  </statusbar>
</overlay>
```

A.1.4 submit.xul

```
<?xml version="1.0"?>
<?xml-stylesheet href="chrome://test_extension/skin/submit.css" type="text/css"?>
<overlay xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul"
      xmlns:html="http://www.w3.org/1999/xhtml" id="test_extension-overlay">
  <!--Invoke JavaScript file: test_extension.js-->
  <script type="application/x-javascript" src="chrome://test_extension/content
      /test_extension.js"></script>
  <window xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul
      " title="Report the webpage">
    <label value="    " />
    <hbox align="center" pack="center" width="100%">
    <!--Hint for user's operation-->
      <label id="reference" value="Evaluate the web page's web service quality
          here:" align="center"/>
    </hbox>
    <hbox align="center" pack="center" flex="1" resizable="false" width="100%"
        >
      <!--Definition of the happy emoticon, also its mouse events-->
      <hbox align="center" pack="center" resizable="false" width="80">
        <image src="chrome://test_extension/content/smileyface.jpg" name="
            jsbutton1" id="imageHappy"
      onmouseover="changeImageURL(this.id, 'chrome://test_extension/content/
          smileyfaceMouseOver.jpg')"
      onmouseout="changeImageURL(this.id, 'chrome://test_extension/content/
          smileyface.jpg')"
      onmousedown="changeImageURL(this.id, 'chrome://test_extension/content/
          smileyface.jpg')"
      onmouseup="window.opener.test_extension.sendRatingInformation(0);window.
          close();"
```

```
20        tooltiptext="I'm Happy" />
21       </hbox>
22       <!--Definition of the unhappy emoticon, also its mouse events-->
23       <hbox align="center" pack="center" resizable="false" width="80">
24         <image src="chrome://test_extension/content/sadface.jpg" name="
               jsbutton2" id="imageSad"
25           onmouseover="changeImageURL(this.id, 'chrome://test_extension/
                 content/sadfaceMouseOver.jpg')"
26           onmouseout="changeImageURL(this.id, 'chrome://test_extension/content
                 /sadface.jpg')"
27           onmousedown="changeImageURL(this.id, 'chrome://test_extension/
                 content/sadface.jpg')"
28           onmouseup="window.opener.test_extension.sendRatingInformation(1);
                 window.close()"
29           tooltiptext="I'm Sad" />
30       </hbox>
31     </hbox>
32   </window>
33 </overlay>
```

A.1.5 test_extension.js

```
1  //a global variable for the reported web page's URL.
2  window.url;
3  var test_extension = {
4      function onLoad() {
5        this.initialized = true;
6      },
7      //Run the extension.
8      function run() {
9                  url = window.top.getBrowser().selectedBrowser.contentWindow.
                       location.href.toString();
10                 window.open("chrome://test_extension/content/submit.xul", "
                       mywindow", "height=170, width=350, resizable=no, top=300,
                       left = 500");

12     },

14     showAbout() {
15         alert("Test Extension, created by HIO");
16     },
17     //redirect to the extension's mainpage, which will be www.yourpagesucks.
           org in the future. Using www.example.com instead temporally.
18     function goHome() {
19         var win = window.top.getBrowser().selectedBrowser.contentWindow;
20         win.open("http://www.example.com");
21         },
22         //
23         function sendRatingInformation(rating)
24         {
25                 var xmlHttp;
26                 try
27                 {
28             // Firefox, Opera 8.0+, Safari
29             xmlHttp=new XMLHttpRequest();
30         }
31         catch (e)
32         {
33         }
34             xmlHttp.onreadystatechange=function()
35                 {
36                   if(xmlHttp.readyState==4) //when the request to the server
                         is done.
37                     {}
```

```
38                  }
39                  xmlHttp.open("GET","http://ms4.vlab.iu.hio.no/qoe.php?rating="
                        +rating+"&url="+url,true);
40                  xmlHttp.send(null);
41          }
42      };
43  window.addEventListener("load", function(e) { test_extension.onLoad(e); },
        false);
44  //a function for changing emoticons' source URL during mouse events
45  function changeImageURL(id,imageURL)
46  {
47          document.getElementById(id).src=imageURL;
48  };
```

A.2 Feedback Collector

A.2.1 qoe.php

```
1   <?php

3   $db = 'qoe';
4   $dbuser = 'qoeuser';
5   $dbpass = 'qoepassword';
6   $dbhost = 'localhost';

8   $link = mysql_connect("$dbhost", $dbuser,$dbpass);

10  if ($link){
11   #       echo "Connection successful!\n<br>";
12      $bfdb = mysql_select_db($db,$link);
13      if ( !$bfdb ){
14          echo "Cannot use $db: " . mysql_error() ."<br>";
15          } else {
16  #           echo "wohoo";
17              $url = $_GET['url'];
18              $rating = $_GET['rating'];
19              $ip = $_SERVER["REMOTE_ADDR"];

21              $string = "insert into qoe_feedback (url,rating,ip  ) values ( '
                    $url',$rating,'$ip')";
22              echo "executing $string";
23              mysql_query($string );
24              echo "OK";
25          }
26  }
27  ?>
```

A.2.2 qoe_search.php

```
1   <?php

3   $db = 'qoe';
4   $dbuser = 'qoeuser';
5   $dbpass = 'qoepassword';
6   $dbhost = 'localhost';

8   $query = $_GET['query'];
9   $since = $_GET['since'];
10  $lastm = $_GET['lastm'];
```

```
12 $link = mysql_connect("$dbhost", $dbuser,$dbpass);

14 if ($link){
15  #       echo "Connection successful!\n<br>";
16     $bfdb = mysql_select_db($db,$link);
17     if ( !$bfdb ){
18         echo "Cannot use $db: " . mysql_error() ."<br>";
19         } else {
20 #           echo "wohoo";
21             $string = "select * from qoe_feedback";
22             if ( $query and $lastm ){
23 #               echo "inserting lastm: $lastm and $query<br>";
24                 $string = $string . " where url like '%$query%' and date > NOW
                       () - INTERVAL $lastm MINUTE";
25             } elseif( $query ){
26 #               echo "inserting query: $query<br>";
27                 $string = $string . " where url like '%$query%'";
28             }

30 #           echo "executing $string";
31             $result = mysql_query($string );
32             echo "<result>\n";

34             while ($row = mysql_fetch_assoc($result)) {
35                 echo "<feedbackitem>\n";
36                 echo "<url>" . $row['url'] . "</url>\n";
37                 echo "<rating>" . $row['rating'] . "</rating>\n";
38                 echo "<ip>" . $row['ip'] . "</ip>\n";
39                 echo "<date>" .$row['date'] . "</date>\n";
40                 echo "</feedbackitem>\n";
41             }
42             echo "</result>\n";
43         }
44 }
45 ?>
```

A.2.3 qoe_result.php

```
1  <?php

3  $db = 'qoe';
4  $dbuser = 'qoeuser';
5  $dbpass = 'qoepassword';
6  $dbhost = 'localhost';

8  $link = mysql_connect("$dbhost", $dbuser,$dbpass);

10 if ($link){
11  #       echo "Connection successful!\n<br>";
12     $bfdb = mysql_select_db($db,$link);
13     if ( !$bfdb ){
14         echo "Cannot use $db: " . mysql_error() ."<br>";
15         } else {
16 #           echo "wohoo";
17             $string = "select * from qoe_feedback";
18 #           echo "executing $string";
19             $result = mysql_query($string );
20             echo "<html>\n";
21             echo "<table>\n";
22             echo "<tr>\n";
23             echo "<td>" . 'url' . "</td>";
24             echo "<td>" . 'rating' . "</td>";
25             echo "<td>" . 'ip' . "</td>";
26             echo "<td>" . 'date' . "</td>";
```

```
27              echo "</tr>\n";
28              while ($row = mysql_fetch_assoc($result)) {
29                  echo "<tr>\n";
30                  echo "<td>" . $row['url'] . "</td>";
31                      echo "<td>" . $row['rating'] . "</td>";
32                      echo "<td>" . $row['ip'] . "</td>";
33                      echo "<td>" .$row['date'] . "</td>";
34                  echo "</tr>\n";
35              }

37              echo "</table>\n";
38              echo "</html>";
39          }
40  }

42  ?>
```

A.3 Web Service Provider: a Munin plug-in

A.3.1 qoe_munin

```
1   #!/usr/bin/perl
2   # setting the query word of:
3   @QUERIES = ( "mysite.org", "playvideo.php" );
4   # The query interval is 5 minutes:
5   $INTERVAL = 5;

7   if ( $ARGV[0] eq "config" ){

9       print "graph_title QoE report plugin\n";
10      print "graph_vlabel Number of unhappy feebacks\n";
11      print "graph_info This plugin shows the number of unhappy feedbacks from
            users using the QoE firefox plugin\n";
12      foreach my $query (@QUERIES ){
13          $query =~ s/\./_/;
14          print "$query.label $query\n";
15      }
16      exit 0;
17  }

19  foreach my $query ( @QUERIES){
20      open(OUT,"wget -q -O - 'http://128.39.73.238/qoe_search.php?query=$query&
            lastm=$INTERVAL' |");
21      my $counter = 0;
22      while ( my $line = <OUT> ){
23          if ( $line =~ /<rating>1/ ){
24              $counter++;
25          }
26      }
27      $query =~ s/\./_/;
28      print "$query.value $counter\n";
29  }
```

Bibliography

[1] http://www.inacon.com/glossary/QoS.php#QoS.

[2] http://compnetworking.about.com/od/networkdesign/g/bldef_qos.htm.

[3] http://gs.statcounter.com.

[4] http://www.getclicky.com/marketshare/global/web-browsers/.

[5] http://www.w3counter.com/globalstats.php.

[6] https://developer.mozilla.org/en/Extensions.

[7] https://developer.mozilla.org/En/Firefox_addons_developer_guide/Technologies_used_in_developing_extensions.

[8] http://en.wikipedia.org/wiki/JavaScrip.

[9] http://www-archive.mozilla.org/xpfe/ConfigChromeSpec.html.

[10] http://en.wikipedia.org/wiki/WAMP.

[11] http://www.webopedia.com/TERM/W/WAMP.html.

[12] http://en.wikipedia.org/wiki/Rating.

[13] http://www.rietta.com/firefox/Tutorial/dist.html.

[14] Firefox. http://en.wikipedia.org/wiki/Firefox.

[15] MySQL. ttp://en.wikipedia.org/wiki/MySQL.

[16] PHP. http://en.wikipedia.org/wiki/PHP.

[17] Quality of service. http://en.wikipedia.org/wiki/Quality_of_service, 2011.

[18] Menasce Daniel A. QoS issues in Web Services. *IEEE Computer Society*, 6, Nov/Dec 2002.

[19] Qinyuan Feng, Yafei Yang, Yan Lindsay Sun, and Yafei Dai. Modeling Attack Behaviors in Rating Systems. In *The 28th International Conference on Distributed Computing Systems Workshops*, pages 241–248, June 2008.

[20] J. Goodchild. Integrating data, voice and video - part 2. Technical Report IP Video Implementation and planning guide, United States Telecom Association, 2005.

[21] David J. Hand Herman J. Ader, Gideon J. MelleBergh. *Advising on Research Methods: A consultant's companion*. Johannes van Kessel Publishing, January 2008.

[22] International Telecommunication Union. *Recommendation E.800: QoS Terms and Definitions related to Quality of Service and Network Performance including dependability*, August 1994.

[23] International Telecommunication Union. *Methods for subjective determination of transmission quality*, August 1996.

[24] International Telecommunication Union. *Perceptual evaluation of speech quality (PESQ): An objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs*, Feburary 2001.

[25] International Telecommunication Union. *Objective perceptual assessment of video quality: Full reference television*, 2004.

[26] International Telecommunication Union. *Mean Opinion Score (MOS) terminology*, June 2007.

[27] International Telecommunication Union. *Recommendation G. 1010: End-user multimedia QoS categories*, November 2010.

[28] Kestutis Mintauckis. Empirical studies of Quality of Experience (QoE) A Systematic Literature Survey. Masterthesis, University of Oslo, April 2010.

[29] Nicholas C. Zakas. *Professional JavaScript for Web Developers,2nd Edition*. Wiley Publishing, Inc., 2009.

[30] Marcio Nieblas Zapater and Graa Bressan. A proposed approach for quality of experience assurance for iptv. In *the First International Conference on the Digital Society ICDS '07)*. Dept. of Computing and Digital Systems Engineering, Escola Politcnica da Universidade de So Paulo, Jan 2007.

[31] N.Sasikaladevi and Dr.L.Arockiam. Reliability Evaluation Model For Composite Web Service. *International Journal of Web & Semantic Technology*, 1(2):16–22, April 2010.

[32] A.S. Singer Patrick, J. Corrie, B. Noel, S. El Khatib, K. Emond, B. Zimmerman, T. Marsh, and S. Nat. A QoE Sensitive Architecture for Advanced Collaborative Environments. In *the First International Conference*

on Quality of Service in Heterogeneous Wired/Wireless Networks. Res. Council of Canada, Ottawa, Ont., Canada, 2004.

[33] Polycom Video Communications. *Quality of Experience and Quality of Service for IP Video Conferencing*, November 2002. white paper.

[34] M. Siller and J. Woods. Improving quality experience for multimedia services by QoS arbitration on a QoE framework. In *the 13th Packed Video Workshop 2003*, Nantes, France, 2007.

[35] Kuan Ta Chen, Chen-Chi Wu, and Wei-Cheng Xiao. OneClick: A Framework for Measuring Network Quality of Experience. In *Proceedings of IEEE INFOCOM 2009*. Inst. of Inf. Sci., Acad. Sinica, Taipei, April 2009.